The Selective Decision Tree Classifier: A Novel Classifier based on Feature Selection

A Thesis Submitted to the
College of Graduate and Postdoctoral Studies
in Partial Fulfillment of the Requirements
for the degree of Master of Science
in the Department of Computer Science
University of Saskatchewan
Saskatoon

By
Jennafer Neiser

Permission to Use

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science
176 Thorvaldson Building
110 Science Place
University of Saskatchewan
Saskatoon, Saskatchewan
Canada
S7N 5C9

Or

Dean
College of Graduate and Postdoctoral Studies
University of Saskatchewan
116 Thorvaldson Building, 110 Science Place
Saskatoon, Saskatchewan S7N 5C9
Canada

Abstract

Living in the era of big data, it is crucial to develop and improve techniques that aid in data processing, such as data reduction. Feature selection is a data reduction technique that generates subsets of data that can be used to build machine learning models. Machine learning utilizes the processing power of a computer to build classification models from the inputted features. By pre-selecting the most relevant features in a data set, machine learning techniques can build simpler and more accurate models.

A novel method of classification based on feature selection is proposed, the Selective Decision Tree classifier (SDTC). The SDTC is derived from the Selective Bayesian classifier (SBC). Both classifiers use Decision Trees (DTs) to rank the features of a data set. DTs are built to organize features in a tree structure, with the most predictive features at the top of the tree, branching down into features that eventually lead to a classification. Features are evaluated and ranked into the levels of the DT. Both classification methods select features from the upper levels of a DT, and those features are fed into a machine learning model built to classify data. The SDTC allows for different depths of levels to be selected, whereas the SBC uses a fixed depth of three levels. To allow for greater scalability of data sets, the SDTC uses DTs as the final machine learning model, whereas the SBC uses the Naive Bayesian model.

The SDTC is applied to three data sets. The first data set is the Level of Service Inventory (LSI) data set that details over 72,000 responses to a recidivism risk assessment test and whether those individuals recidivated. The second data set is provided by the Saskatoon Police Service and contains information on missing children cases. The third data set provides the scores and reaction times recorded from the Computerized Assessment of Mild Cognitive Impairment (CAMCI) test taken by individuals as a pre-diagnostic tool for Alzheimer's disease and dementia. To demonstrate where the SDTC excels at building better predictive models, two other classification models were built from the data sets, DTs and the SBC.

When comparing the SDTC to a DT model, the advantages of feature selection are clearly evident as demonstrated by the improved accuracy. Using the LSI data set, the SDTC selected a single feature out of 43 and obtained over a 76% accuracy, compared to the 71% accuracy obtained by a DT model using all the features. The highest accuracy, 92%, was seen using the SDTC on the Missing Persons data set, almost a 7% increase using only two of the 90 features when compared to DTs. Using the CAMCI data set, the SDTC achieved a 60% accuracy using only 12 of the 33 features, compared to a 58% accuracy obtained by DTs using all the features. The flexibility of the SDTC also gave it distinct advantages and in some cases improved accuracy in its predictive potential when compared to SBC. The SDTC outperformed the SBC on two of the three data sets. The highest increase in accuracy when comparing the SDTC and the SBC was obtained using the Missing Persons data set; the SDTC achieved 92% accuracy, a 7% higher accuracy than the SBC. The demonstrated capability of the SDTC supports its potential for analyzing a variety of data sets and obtaining a deeper understanding of how DT structures can be used for feature selection.

Acknowledgements

I would like to acknowledge the support of the Computer Science Graduate Studies Department for their constant help and knowledge provided. I would like to acknowledge my parents for the support they provided me through constant love and words of perseverance. I would like to thank my husband, Brandon, for his constant patience and encouragement during the many hours of writing and research. I would like to thank my supervisor Dr. Raymond J. Spiteri. I would also like to recognize the agencies that provided me and the team I work with ethical access to the data worked with in this thesis: Ontario Ministry of Community Safety and Correctional Services, Saskatoon Police Services, Psychological Software Tools, Inc. (Pittsburgh, PA), the Psychology Department at the University of Saskatchewan, and the Centre for Forensic Behavioural Science and Justice Studies at the University of Saskatchewan.

Contents

List of Tables

List of Figures

List of Abbreviations

AD	Alzheimer's Disease
AUC	Area Under the Curve
CAMCI	Computer Assessment of Mild Cognitive Impairment
CART	Classification and Regression Tree
CT	Classification Tree
DT	Decision Tree Classifier
LSIT	Latest-Substring Index Tree
LSI	Level of Service Inventory
LS/CMI	Level of Service/Case Management Inventory
LSI-R:SV	Level of Service Inventory - Revised: Screening Version
MCI	Mild Cognitive Impairment
MMSE	Mini Mental State Examination
NBD	Naive Bayes with Discretization
NBK	Naive Bayes with Kernel Density Estimation
RBFN	Radial Basis Function Network
SBC	Selective Bayesian classifier
SDTC	Selective Decision Tree classifier
SPECT	Single Photon Emission Computed Tomography
SPS	Saskatoon Police Service
SVM	Support Vector Machine

1 Introduction

Transitioning from the data explosion era to the big data era leaves many new and ever-changing problems for researchers to face (*Cai and Zhu*, 2015). Data scientists are no longer working with data units of GB or TB, but larger units like PB (1 PB = 2^{10} TB), EB (1 EB = 2^{10} PB), and ZB (1 ZB = 2^{10} EB). A major challenge when working with big data is ensuring the quality of the data and consequently any models built with the data. The more data under consideration, the harder it is to judge data quality within a reasonable amount of time (*Cai and Zhu*, 2015). Data analytics plays a part in the processing of big data to improve the data quality. By improving data quality, the efficiency of data utilization will increase, and the risk of poor classification models being built and used will decrease.

When working with a data set that has many features (high dimensionality), there can be features that do not add to the predictive power or accuracy of a model. Features that do not contribute to the overall accuracy of a model are redundant and ultimately unneeded. Neglecting to remove irrelevant or redundant features can lead to over-fitting and longer training times for machine learning models (*Brownlee*, 2016b). Also, working with smaller groups of selected features often leads to simpler models that are easier to interpret. Accordingly, feature selection is often an important process to perform on big data sets to help improve the quality of the data for the classification models being built.

This thesis explores a novel classification method based on selecting the most important features from a data set: the Selective Decision Tree classifier (SDTC). The SDTC is derived from a technique called the Selective Bayesian classifier (SBC) (*Ratanamahatana and Gunopulos*, 2002). The SDTC first uses Decision Trees (DTs) to determine which features of a data set are the most important. Using only these selected features, a DT is applied a second time to build a final classification model. The goal is that minimal prediction accuracy is lost and the number of features used to build a classification model is reduced. By achieving this goal over multiple data sets, the argument that SDTCs are a valid way to improve data quality is strengthened. Also by applying the SDTC to the three data sets used in this thesis, advances in usable prediction models and feature understanding can be made in those areas of study.

1.1 Machine Learning

Since the beginning of the computer era, machine learning has been used to discover facts and theories about data by enhancing predictive tests (*Michalski et al.*, 2013). There are many trusted machine learning methods that have been developed, e.g., DTs (*Quinlan*, 1993, 1986), Naive Bayesian classification (*Elkan*,

1997), and Support Vector Machines (*Gunn et al.*, 1998). Machine learning methods can be used to build models that classify data; those classifications can be interpreted to inform predictions. DTs are the main machine learning method used in this thesis. A DT is built using a set of data consisting of features; the features are layered into a tree structure that can be used for classification. Where the features are placed in the tree is determined by a splitting criterion. For example, consider a data set that represents a series of responses to a set of questions used to determine whether or not an individual might commit a crime. The questions become the layered features of the DT, and the responses to the questions are the data used to build the DT. The DT model classifies whether or not an individual is likely to commit a crime.

1.2 Feature Selection

Feature selection is the process of eliminating features or dimensionality in a data set while still retaining the accuracy of a model built with the entire data set. Often there are features in large data sets that do not provide meaningful information and can be considered redundant. Removing these features reduces the chances that the model built is over-fit to the data used (*Brownlee*, 2016b). A model is over-fit when the performance of the model built with the testing data declines greatly when used to classify testing or real world data (*Cawley and Talbot*, 2010). The feature selection method, SDTC, uses DTs to both select features from a data set and to build a final classification model. The SDTC is derived from the SBC (*Ratanamahatana and Gunopulos*, 2002).

1.2.1 Selective Decision Tree Classifier

The SDTC uses DTs for ranking and selecting features and for building the classification model. The SDTC refines the initial steps of the SBC and replaces the final model with a DT. In this thesis, the DT steps were tested with both Gini and Entropy DTs (*Buntine and Niblett*, 1992; *Quinlan*, 1993). The difference between a Gini and Entropy DT is that they use different splitting criteria to determine where features are placed in the DT, resulting in different models being built. These two DT splitting criteria have some of the lowest error rates when compared to other DTs using different splitting criteria tested over multiple data sets (*Buntine and Niblett*, 1992). In this thesis, the results obtained by both DTs exhibited little variation in accuracy, but there was some variation in the features selected.

The SDTC was tested using three different data sets. The results obtained from the SDTC were compared to the results of a DT built with the three data sets. Comparing the results from DTs and the results from SDTCs gives an indication of how well the feature selection method works and whether it is beneficial to use. The SDTC was also compared to the SBC. This comparison shows where the SDTC is an improvement from the SBC and where they perform similarly. The rest of this chapter briefly describes the data sets and highlights the results obtained.

1.3 Data Sets

Multiple data sets are used to determine the capabilities of the SDTC. The first data set is comprised of recidivism risk assessment scores from the Level of Service Inventory (LSI) test. The second data set is comprised of data collected by the Saskatoon Police Service (SPS). The final data set is comprised of test scores from the Computerized Assessment of Mild Cognitive Impairment (CAMCI).

1.3.1 LSI

In this thesis and generally speaking, recidivism is referred to as the tendency of a convicted criminal to re-offend. Recidivism risk assessment tools are designed in an attempt to take into consideration the many variations in human behaviour when predicting the likelihood of an individual to re-offend (*Andrews and Bonta*, 2010). The *Level of Service/Case Management Inventory* (LS/CMI) recidivism risk assessment tool was used in Ontario, Canada, from 2010 to 2011, and the results were recorded for 72,726 individuals. These records make up one of the data sets used in this thesis. The LS/CMI is known generally as the LSI tool because it is one of many variation of the tool. The LS/CMI is comprised of 43 yes/no questions that are used to determine a risk score (*Andrews et al.*, 1995). The higher the score obtained with the LSI tool, the greater the risk of an individual to re-offend. Questions in the data set are referred to as *features* and have a binary value of 1 for yes or 0 for no. The SDTC is applied in an attempt to reduce the number of features that are required to accurately provide a risk score.

The 43 LSI features are grouped into eight categories: Criminal History (8 features), Education/Employment (8 features), Family/Marital (4 features), Leisure/Recreation (2 features), Companions (4 features), Substance Abuse (8 features), Pro-Criminal Attitudes (4 features), and Antisocial Pattern (4 features) (*Andrews and Bonta*, 2010). The number of features in each category can be interpreted as the importance of that category; i.e., a larger number of questions in a category correlates to a heavier impact that category has on the total score.

1.3.2 Missing Youths

SPS is in the process of implementing the Saskatchewan Police Predictive Analytics Lab (SPPAL) in partnership with the Ministry of Corrections and Policing and the University of Saskatchewan. The purpose of this lab is to aid the officers and community safety partners in their normal duties by providing user friendly predictive analytics tools to them. The lab is currently working on an inaugural project on Missing Persons. The data provided by the SPS contained 434 missing youths cases with 91 features for each case. The data used with the SDTC were obtained from the provided Missing Youths Database located in the SPPAL.

1.3.3 CAMCI

The CAMCI is a mild cognitive impairment test administered as a computerized tool. A patient performs a series of tests on a tablet while the program records scores and reaction times for each of the tests (*Saxton et al.*, 2009). CAMCI is a computerized version of paper-and-pencil tests combined with a virtual reality shopping trip where the patient must remember a series of tasks. CAMCI is intended for older individuals who have not been previously diagnosed with a mild cognitive impairment. In total, the test takes on average 1484 seconds or 24.73 minutes to complete; a long time for a clinician to administer a test to a single person. The goal is to derive the key features of the CAMCI test to allow for a shorter pre-diagnostic tool.

The data set used contains 33 features and 580 cases. The data are obtained from Psychological Software Tools, Inc. (Pittsburgh, PA), and the participants were adults living in western Pennsylvania. The features in the data set consist of the reaction times and scores obtained for each test the participant took. There are also features that record the participants age, schooling level, and gender.

1.4 Ethics

Some of the data in this thesis are highly sensitive, and great care was taken to uphold a high ethical standard. The results given in this thesis do not divulge aspects of the data that could be seen as sensitive or proprietary. The University of Saskatchewan ethics file that contains the necessary permissions to work with the data is BEH# 16-166.

1.5 Contributions

The previously discussed data sets were used to test the SDTC. The results obtained from the classification models were interpreted, and the potential advances in the various fields of study were stated. The rest of this chapter briefly discusses the results and how they can benefit the fields of study to which they pertain.

1.5.1 LSI

Working with the data set of LSI scores, the SDTC was used to identify the crucial features, creating a reduced question set. Reducing the question set can improve the ease of tool administration and potentially allow for the re-allocation of resources to individuals according to their risk of recidivism. For example, being able to provide higher-risk individuals with more resources, such as rehabilitation counseling, can lower the chance of an individual recidivating. Another potential advantage is that removing less relevant data can lead to more efficient computations, saving time and resources. A further potential advantage for researchers is that building a model with a smaller set of features can result in the model being simpler, thus easier to understand (*Brownlee*, 2016a). With a simpler model, a greater number of unspecialized people would be able to interpret the model and how it can be used.

It was found that there is no loss in accuracy when the SDTC is used compared with the a DT model built with all 43 of the LSI features, demonstrating that the SDTC is a valid way to select features. Using SDTCs, the set of features used to build the model can be reduced to a single feature, for which an accuracy of 76% is obtained, an increase of over 4% from the DT model. The SDTC model also obtained a 33% sensitivity and a 95% specificity, compared to the model built with all features that obtained a 24% sensitivity and a 92% specificity. It is interesting to compare the features selected by the various frameworks of the SDTC tested, as there are many that overlap. Looking at these features, conclusions can be drawn regarding the importance of the features that overlap in the data set.

1.5.1.1 LSI Screening Test

The LSI recidivism risk assessment test has a short form version called the Level of Service Inventory - Revised: Screening Version (LSI-R:SV) used to screen out individuals who are least likely to re-offend (*Andrews and Bonta*, 1998). LSI-R:SV contains a subset of 9 questions or features from the original 43. If an individual scores from 0 to 2 they are at minimum risk, if they score from 3 to 5 they are at medium risk, and if they score from 6 to 8 they are at maximum risk. At minimum risk the full LSI is desirable, at medium risk the full LSI is strongly recommended, and at maximum risk the full LSI is mandatory (*Andrews and Bonta*, 1998). This test was put together theoretically, but it has not been extensively tested and validated. By comparing the accuracy, specificity, and sensitivity of the SDTC and the LSI-R:SV, the LSI-R:SV can be validated, confirming the features it contains are the most well-suited for the task, or it can be improved using other features selected by the SDTC.

Using the all of the LSI-R:SV features to build a DT, an accuracy of 74% was obtained. The best model built using SDTCs achieved a slightly lower accuracy of 73%. The sensitivity of the LSI-R:SV model was 35% compared to the 15% SDTC obtained. The specificity of the LSI-R:SV model was 92% compared to the 99% SDTC obtained. The features used in the SDTC models and LSI-R:SV varied greatly.

1.5.2 Missing Youths

With feedback and input from the officers at SPS, two questions were asked of the missing youths data. The first being which children, once found, are likely to run away again. Habitual runaways or missing youths make up a majority of the missing youth cases the officers at SPS encounter. The second question asks whether or not a missing child is likely to be involved with gang activity. With answers to these two questions, officers may be able to have a deeper insight into how to approach new missing youths cases and finding the missing children.

Using the SDTC, features were extracted and used to build a predictive model that gives Missing Persons officers possible answers to their questions. The SDTC models built for both questions out performed the DT models built with all 91 features in the data set. For question one, the SDTC model obtained a 92% accuracy, a 79% sensitivity, and a 100% specificity compared to a DT built with all features that obtained a

84% accuracy, a 73% sensitivity, and a 92% specificity. For question two, the SDTC model obtained an 88% accuracy, a 50% sensitivity, and a 95% specificity compared to a DT built with all features that obtained a 83% accuracy, a 50% sensitivity, and a 90% specificity.

1.5.3 CAMCI

The SDTC was able to obtain a higher predictive accuracy than regular DTs on the CAMCI data set. Each component of the CAMCI test takes different times to complete; the goals were to reduce the total completion time by selecting a subset of components, and to increase the accuracy of the test. To accomplish this, the average time associated with each component was used to determine the length of the test. Each component or piece of the test may have multiple features associated with it, i.e., score or reaction time. If a feature of a component was selected, the average time to complete that component was added to the total time it would take to complete the CAMCI test. It was found that the accuracy could be improved by almost 3% while reducing the time of completion by 3 minutes. The shortest completion time was 14 minutes, reducing the completion time by over 10 minutes, while still increasing the accuracy by 1%. The accuracy of the best SDTC was 59% compared to a DT built with all features that obtained a 56% accuracy. The best SDTC obtained a 75% sensitivity and a 39% specificity. A DT built with all the CAMCI features obtained a 70% sensitivity and a 39% specificity.

1.6 Outline

This thesis contains the research methods used and results obtained when using the SDTC on the LSI data set, the missing youths data set, and the CAMCI data set. Chapter 2 is a summary of other papers that consist of important information to comprehend aspects of this paper. Chapter 3 is a description of the machine learning methods used, including DT and SDTC. Chapter 4 is a write up of the procedures followed to obtain the given results, a detailed description of the data sets used, and a write up and interpretation of the results and how they are beneficial to the various fields the data relates to. Chapter 5 is a summary of the final results and future work.

2 Literature Review

This section introduces concepts and background information associated with each of the main topics permeating this thesis. Section 2.1 discusses machine learning topics as a whole and how they have been developed to enhance predictive tests. Section 2.2 discusses feature selection and the SDTC method. The SDTC method combines machine learning techniques with feature selection to produce a classification model. Section 2.3 provides background information about recidivism risk assessment tools and how machine learning has been used to improve them. Section 2.4 provides background information on the techniques that Missing Persons officers and officials currently use to solve missing persons cases, including missing youth cases. Section 2.5 provides background information on how machine learning has been used to improve cognitive impairment pre-diagnostic tests.

2.1 Machine Learning

Machine learning methods are designed to build models used for the classification of data. Classifications can be used to aid in predictions or to determine likely outcomes of scenarios. There are a vast number of data sets and machine learning methods that can be combined to build models that provide classifications. There are many possible combinations of methods and data sets, but not all will suit an individuals needs or have a good predictive accuracy. This section provides applications for some commonly used machine learning methods and how the machine learning methods performed.

Machine learning methods can be used to build classification models and understand a variety of problems. For example, one problem is to build a model that classifies whether a product review is positive or negative (*Pang et al.*, 2002). A study done by *Pang et al.* employed a model developed from words that a human intuitively assumes are present in good or bad product reviews. This model proved to be poor, obtaining only a 64% accuracy and resulted in many product reviews being classified as ties, both good and bad. This human model was then compared to a basic model built from frequency counts of the known reviews (*Pang et al.*, 2002). The frequency counts model was able to achieve a higher accuracy and fewer ties, validating that machine learning had potential to greatly improve the accuracy of human-based classification. Next, three machine learning models were built and compared to determine which had the best accuracy on the data being analyzed (*Pang et al.*, 2002). It was found that from the three models built, Support Vector Machines (SVMs) performed the best, achieving the highest accuracy. A Maximum Entropy model achieved the next highest accuracy, and a Naive Bayesian model performed the worst. *Pang et al.* did note that even

though SVMs performed the best, the difference in accuracy between SVMs and the other two methods was small.

In (*Nguyen and Armitage*, 2008), different machine learning methods are applied and other literature was surveyed in an attempt to build models that classify Internet traffic. All machine learning methods sort or prioritize features differently, resulting in a variation of outcomes and predictions (*Nguyen and Armitage*, 2008). *Nguyen and Armitage* report the findings of *Williams et al.* where a variety of supervised machine learning methods were applied including Naive Bayesian with Discretization (NBD), Naive Bayesian with Kernel Density Estimation (NBK), C4.5 DT, Bayesian Network, and Naive Bayesian Tree. It was found that all methods but NBK achieved greater than 95% accuracy. Further, it was found that C4.5 DT is the fastest computationally, followed by NBD. Feature reduction or selection was also greatly improved the performance. *Nguyen and Armitage* also reported the findings of *Erman et al.*. It was found that when comparing AutoClass, a clustering technique, to Naive Bayesian classification, AutoClass performs better than Naive Bayesian in terms of precision and recall. Computationally, Naive Bayesian was significantly faster when compared to AutoClass. Naive Bayesian took 0.06 seconds compared to the 2070 seconds reported for AutoClass. Overall, *Nguyen and Armitage* found that when used properly, many machine learning methods have the potential to build successful models that can correctly classify Internet traffic. A second point that *Nguyen and Armitage* noted was that DTs performed well across many of the surveyed studies.

DTs are a predominant machine learning method used in this thesis, and therefore it is important to note where they excel. There are three areas identified as key characteristics of machine learning methods, and DTs excel in all of them; the strategies used, the representation of knowledge, and the application domain of the system (*Carbonell et al.*, 1983). DTs are used to classify data and answer questions using a variety of strategies; Entropy and Gini splitting criteria being two of the most popular strategies. The way a DT model is built embeds the knowledge used to build it into the structure of the final model. A DT is built top down, guided by the repetition of patterns in the data used to train and build the model (*Quinlan*, 1986). The structure consists of a series of features, starting with the root of the tree, or most significant feature, and branching down until a final classification is reached. DTs can be used on a wide variety of data sets to answer different questions from many fields of study, giving them a wide application domain. A detailed example and explanation of how a DT is built is found in section 3.2.

The following two subsections highlight elements of machine learning that are important to understanding this thesis. Section 2.1.1 reviews the importance of data cleaning. Section 2.1.2 reviews the Random Forests machine learning method (*Breiman*, 2001; *Liaw et al.*, 2002). The SDTC is similar to this method, but it is important to note the key differences.

2.1.1 Data Cleaning

There is no shortage of data in the world, from online sources to data collected privately, and as the world advances more of the data sets collected are becoming large (*Cai and Zhu*, 2015). When working with large

amounts of data, it is important that all portions of the data add information or have relevance to the data set and the model being built, ensuring *data quality*. Data quality can be ensured in numerous ways, some of which involve the removal of redundant data or adding missing pieces of important data. There are many instances where data quality has proved extremely valuable in different areas of study (*Batista and Monard*, 2003; *Yang et al.*, 2003). The remainder of this section reviews studies where this fact is demonstrated.

In (*Batista and Monard*, 2003), the benefits of treating missing data were explored. Without the proper treatment of missing data, unwanted biases can be added into the data. *Batista and Monard* explored missing data treatment techniques on four distinct data sets. The majority of data sets did not initially have missing data, and the few items that were missing from the initial data sets were removed entirely. Missing data points were then randomly input at different percentages for the data sets. The missing values were treated using mean or mode imputation, the C4.5 DT and CN2 methods, and k-nearest neighbors to replace or substitute the missing data (*Batista and Monard*, 2003). It was found that using k-nearest neighbors to replace the missing values yielded lower error rates and was beneficial when compared to not using a method to improve the data quality. There were some limitations found when replacing missing data points. One limitation being, if the missing data have similar information to other data fields already present, filling in the missing data can be useless or even harmful to the prediction capabilities of the model (*Batista and Monard*, 2003).

In (*Yang et al.*, 2003), Web-log data are used to build a model that can predict a user's actions based on their past actions. Building a model to predict the behaviours of people can be highly complex, and because of this, a method to build a concise and clean model is required. *Yang et al.* used an association rule-based model to predict the next actions a user will take on a Web browser. To ensure the usability of this model, redundant information was removed from the prediction model using an algorithm that was developed called Latest-Substring Index Tree (LSIT) (*Yang et al.*, 2003). When using web data, there are many data points that are unneeded but recorded some being image requests or documents not requested directly by the user. By using the LSIT data quality enhancing method, *Yang et al.* were able to condense the classification model without decreasing its accuracy. The method of selecting the critical pieces of a data set and disregarding the rest while still keeping the accuracy of a classification model is called *feature selection*. Section 2.2 gives a deeper understanding and more evidence that feature selection is a useful and beneficial tool when building classification models with machine learning.

2.1.2 Random Forests

A random forest uses DTs to build a classification model (*Breiman*, 2001). A set of independently constructed DTs are built using a bootstrap sample method of the data set (*Liaw et al.*, 2002). These DTs are not built using the best splitting feature among all features like a typical regression DT. The best splitting feature is determined from a randomly selected subset of the data's features. The final classification is determined by a majority vote of all the DTs constructed (*Breiman*, 2001). The two aspects of a random forest that contain

variability are the number of DTs in the forest and the number of features to be considered when obtaining the best splitting feature. Since their introduction by *Breiman* in 2001, random forests have been applied to many areas to aid in classification and feature analysis (*Tribby et al.*, 2017; *Melnychuk et al.*, 2017).

A study analyzing a person's choice of walking route utilized random forests to build a model that dramatically improved the achievable results compared to previously built models (*Tribby et al.*, 2017). The GPS data used by *Tribby et al.* were first processed to understand when trips were taken, the route, and other features. The final data set contained many features (*Tribby et al.*, 2017). The random forest technique was used to select features that were important to an individual choosing a walking route and compared them to features in a well-known theory-driven method. Twenty features were selected using the random forest method, whereas the theory-driven method used six features. The greater number of features used by the random forest method allowed researchers to more deeply understand how people chose walking routes. The random forest showed that a walking route is more likely to be chosen if it has offices, graffiti, and on-street parking. It was noted by the researchers that "these variables appear to be proxies for unmeasured variables" (*Tribby et al.*, 2017).

The way fisheries are managed has a large impact on the fish population, but it is unclear which aspects of the many management systems have the greatest impact (*Melnychuk et al.*, 2017). The research done by *Melnychuk et al.* used random forests to help assess which attributes of fishery management systems contribute the most to the fish population. It was found that there were three main variables that made a positive impact. The first had to do with the wealth of the country the fisheries were located, the more wealthy the better the fishery management and fish population. The second pertained to the volume of fish caught, the more fish caught the more resources that are used to better the fisheries. The third feature has to do with the resources and money the country invests into the fisheries. Random forests allowed *Melnychuk et al.* to determine important features that impact the management and success of the fisheries.

Random forests are similar to the novel method, SDTC, introduced in this thesis. It is important to highlight the key differences between the two methods to ensure the SDTC technique makes changes to the already developed random forest technique. Random forests and SDTCs both use multiple DTs in their algorithms, both with the ultimate goal of feature selection and classification. They key difference in the feature selection aspect is that the SDTC selects features according to the depth level in a built DT, utilizing the structure and the way features are ranked in a DT as a whole. In contrast, random forests use the best splitting feature of a random subset of the original feature set when building each DT in the forest. The other key difference between SDTC and random forests is the way they classify data. SDTCs build a final DT model using the selected features it obtained in the first step of the algorithm. In contrast, random forests take a majority vote from the many DTs built in the model (*Breiman*, 2001). The two methods try to accomplish the same goal of feature selection and classification using DTs but in different ways.

2.2 Feature Selection

Feature selection is an important step in preparing and ensuring the quality of the data to be used by a machine learning method to build a model. The irrelevant and redundant features are removed from consideration and the most relevant features are left. Feature selection has a wide field of application, from genetics to diagnosing diseases to text mining (*Salas-Gonzalez et al.*, 2010; *Li et al.*, 2004; *Mugunthadevi et al.*, 2011; *Chandrashekar and Sahin*, 2014).

Genes are complex structures that contain information about the processes of a human body. The cell's job is determined by the production of proteins made up of amino acids. *Li et al.* studied these cells and their structures to develop a feature selection and classification method for tissue classification. Tissues have multiple classes and can be challenging to classify because of this. The difficulty is due to the fact that the data are highly dimensional and the sample size of the data set is generally small. Due to the high dimensionality of the data, feature selection is important. The Rankgene program (*Su et al.*, 2003) was used, allowing for eight feature selection methods to be tested: information gain, twoing rule, sum minority, max minority, Gini index, sum of variances, one-dimensional SVM, and t-statistics. The results showed that there was no feature selection method that outperformed the rest (*Li et al.*, 2004). It was found that over various combinations of data sets, machine learning models, and feature selection methods, feature selection does improve the accuracy of the machine learning model being used (*Li et al.*, 2004). How the machine learning model performed in conjunction with the feature selection method seemed to be complicated and hard to explain on a general level. It was found that the choice of the machine learning method had a larger impact than the feature selection method on the accuracy of the classification model (*Li et al.*, 2004).

Text mining is used to sift through massive collections of documents and extract meaningful information. Feature selection is a clear tool to aid in this process. *Mugunthadevi et al.* explored how feature selection methods perform when applied to text mining problems. The goal of their study was to cluster documents into groups that share a similar topic. Using feature selection reduces the dimensionality of the feature space and provides a better understanding of the data, improving the clustering result (*Sebastiani*, 2002). *Mugunthadevi et al.* report on work by *Zamir and Etzioni* where an object-rich subtree is developed to locate irrelevant objects that are then eliminated. With the remaining features, a directed graph was built. The directed graph placed nodes describing classes of structurally similar pages and branches between the pages. *Mugunthadevi et al.* documented that *Xu et al.* put forward a new feature selection method that utilized expectation maximization, cluster validity, and Davies–Bouldin's index. The report indicates clear advantages to this approach over human categorization. Feature selection is a tool that gives clear advantages to text mining problems (*Mugunthadevi et al.*, 2011).

A survey on feature selection methods was done by *Chandrashekar and Sahin*. A multitude of feature selection methods from filtering methods to technical algorithms were discussed and defined. After weighing pros and cons, a modified Genetic Algorithm called CHCGA (*Eshelman*, 1991) was chosen as the feature

selection method to be used. After the features were selected, two different classifiers were used, SVM and a Radial Basis Function Network (RBFN, a feed-forward neural network). Using seven different data sets to test their method, it was found that using CHCGA and SVM performed better than CHCGA and RBFN on five of the seven data sets. Overall, it was found that feature selection provides evident benefits when building classification models, such as providing insight into the data, more accurate classifier models, enhanced generalization, and the identification of irrelevant variables (*Chandrashekar and Sahin*, 2014).

The feature selection technique presented in this thesis is the SDTC derived from the SBC. SBC uses Naive Bayesian as its final machine learning model, whereas SDTC uses DTs as its final model. The following Section 2.2.1 details the feature selection technique SBC and the experiments done by *Ratanamahatana and Gunopulos*.

2.2.1 Selective Bayesian classification

The SBC is a feature selection and machine learning method developed by *Ratanamahatana and Gunopulos*. The goal of this method is to improve the capabilities of the Naive Bayesian classifier, which performs poorly on data that has correlated features. The SBC shuffles the given training data set, and selects a 10% sample to build a DT model. Using the built DT model, it selects the top three layers of features as relevant. The SBC repeats this process five times and builds a final Naive Bayesian classifier using the relevant features selected. A more detailed description of this process can be found in Section 3.5.

The SBC was tested on ten different data sets. For each data set various splits of training and testing were tested for three different classification methods, Naive Bayesian, an Entropy DT, and SBCs itself. It was reported that SBCs did better than the other methods tested in most circumstances. DTs performed better than SBC on three of the ten data sets and Naive Bayesian did not outperform SBC on any of the data sets. It was found that when the features that are closer to the root of an Entropy DT are used to build a Naive Bayesian model, it outperforms a Naive Bayesian model built with all the features (*Ratanamahatana and Gunopulos*, 2002).

There are some limitations to the method that *Ratanamahatana and Gunopulos* developed. This study proposes methods to fix these limitations. Looking back to the results presented, it was found that DTs did outperform SBCs on three data sets. This led to the idea that using DTs as the final model in place of the Naive Bayesian classifier could lead to improvements in accuracy of the final model. Another limitation of this study was that the way the features were selected from the DT was fixed. It was reported that the features selected were those present in the first three layers of the DTs. Adding variation to the depth level of layers selected could lead to a more customizable algorithm. This customization capabilities could lead to an algorithm that is more successful on varying data sets. Lastly, the way the training data are split up to create the five DTs allows for overlapping data to be used to build them, and all data in the training set are not utilized. The change proposed is to split the data into five distinct pieces, building a DT from each split of data. All of these changes to SBCs are implemented in SDTCs, the method explored in this thesis.

2.3 Recidivism Risk Assessment Tools

Criminal justice research in Canada focuses on four key principles: risk, need, responsibility, and professional discretion (*Correctional Service of Canada, Policy and Research Sector, Research*, 2015; *Andrews et al.*, 2006). Retaining theses four principles helps to ensure that Canadian researchers do not turn the field of criminology into "one preoccupied with the art of punishment and the science of oppression" (*Correctional Service of Canada, Policy and Research Sector, Research*, 2015). Recidivism risk assessment tools are one aspect of using algorithms and statistics to aid in this endeavor. There are many versions of recidivism risk assessment tools, some using machine learning methods and others built theoretically by psychologist, criminologist, and correctional professionals (*Andrews and Bonta*, 2010; *Ting et al.*, 2017; *Tollenaar and Van der Heijden*, 2013; *Ozkan*, 2017).

In Singapore, a large amount of recidivism risk assessment data was collected by probation officers. *Ting et al.* developed a model from the data to help classify youth offenders by the probability of re-offense. The classified higher risk youth offenders would then be provided rehabilitation efforts. The model built from the data consisted of a large collection of classification trees called a random forest model. A random forest is a classifier made up of decision trees, this is called an ensemble classifier (*Breiman*, 2001; *Zhang et al.*, 2003). The initial data was split into 60% training and 40% testing to help ensure the model was not over-fit to the data. A model that is over-fit to data demonstrates high predictive capabilities that cannot be reproduced in the real world. The accuracy of the random forest model was 65%. The area under the curve (AUC) score obtained by the built model was 0.69 where a logistic regression model only achieves an AUC score of 0.62 (*Ting et al.*, 2017). The findings illustrate that machine learning models are competitive with traditional conventional statistics (e.g., logistic regression) (*Ting et al.*, 2017).

Tollenaar and Van der Heijden surveyed whether a statistical, machine learning, or data mining classification model predicts recidivism best. There are two types of features that can be used to build a model, static or dynamic. Static features do not change over time, or change in one direction (like age), and are used in actuarial risk assessments. Dynamic features can change dramatically overtime, for example having an education, or the type of friends an individual associates with. Using three sets of data (general recidivism, sexual recidivism, and violent recidivism), eleven models were tested and compared using a combination of static and dynamic features. It was seen that on general recidivism data, logistic regression (*McCullagh and Nelder*, 1989) performs marginally better than the other models, and neural networks do best in terms of calibration error of the model. For sexual recidivism, the linear discriminant analysis model (*Fisher*, 1936) seemed to perform the best. For violent recidivism, there were a few models that performed well: logistic regression, adaptive boosting (*Freund and Schapire*, 1997), and partial least squares (*Wold*, 2004). The conclusion was that there is no model that always performs better than the rest, and that the model that may perform best in one situation may not perform well in another (*Tollenaar and Van der Heijden*, 2013).

Ozkan studied the performance of various machine learning methods applied to a set of recidivism data.

The initial step in the research was to select a set of features that build a good predictive model. The features were selected by removing the highly correlated features and then applying the Least Absolute Shrinkage and Selection Operator methods (*Tibshirani*, 1996). The final set of data used to build the models contained 80 features. The models tested were logistic regression, random forest, SVMs, XGBoost, neural networks, and search algorithms. Three different models obtained the best accuracy, sensitivity, specificity, AUC, false negative rate, false positive rate, and precision. XGBoost scored highest in accuracy and AUC. Support vector machines scored the highest in sensitivity and false negative rate. Logistic regression had the highest score in precision, specificity, and false positive rate. The paper concluded that logistic regression is sufficient for being able to perform adequately across all metrics, but ultimately all the methods scored closely, and that from one data set to another, the best model may change (*Ozkan*, 2017).

2.3.1 LSI

The Level of Service Inventory (LSI) risk assessment tool is used to understand and measure the risk at which a criminal is to re-offend (*Andrews et al.*, 1995). The tool is administered to individuals upon release from custody and at designated time intervals thereafter. The LSI tool is comprised of forty-three questions that are also referred to in this thesis as *features*. Each question has a response of "yes" or "true" and assigned a value of 1, or "no" or "false" and assigned a value of 0. These values are then totaled to produce an LSI score. The higher the LSI score, the higher the risk of recidivism. The score of 22 is the value at which the rate of recidivism crosses the 50% threshold. Accordingly, individuals with scores of 22 or higher are considered to be at risk to recidivate. Versions of the LSI tool have been used for more than twenty-five years, with increasing popularity in the recent years (*Wormith*, 2011). With this increase in use, it is crucial to ensure the tools have as much predictive accuracy as possible while being a reasonable length to facilitate practical administration. The LSI has been studied for predictive validity, but few machine learning methods have been applied and tested (*Girard and Wormith*, 2004; *Oraji*, 2016).

Girard and Wormith studied the Level of Service Inventory - Ontario Revision (LSI-OR) for predictive validity. LSI-OR is an empirically and theoretically developed test (*Andrews et al.*, 1995). The LSI-OR test was administered by classification officers or community probation officers for the study. The total number of offender records and LSI-OR tests administered was 698. After removing female and young offenders from the data set because of their small sample size there were 630 valid data entries. The recidivism rate of the data set consisting of only adult male offenders was 54.4%. The AUC score of the LSI-OR test for general recidivism was 0.70. It was found that the predictive validity of the test was similar to other case studies, and the findings supported the use of this version of the LSI test.

Oraji studied the LSI-OR data and applied the Naive Bayesian machine learning method. The data set used had 72,725 records with 48 features for each of them. The study focused on improving the predictive capabilities at the mid range scores of the LSI-OR test. The mid range scores predict recidivism with close to 50% accuracy, a score showing the unreliability of the predictive test. Applying Naive Bayesian classification

allowed for a slight increase in accuracy for these mid range scores. Looking at the capabilities of Naive Bayesian and the LSI-OR test, the accuracies able to be achieved are very similar, 75% for LSI-OR and 74% for Naive Bayesian.

2.4 Missing Persons

A missing person can refer to an individual who has been missing for a few hours to someone who has been missing for months and is presumed dead. In the context of this thesis, a missing youth or person is an individual who has just been reported missing and who is presumed to be alive. When children in particular go missing, it can be a traumatic and intense situation. Understanding why children run away or go missing is key. *Bonny et al.* analyzed the characteristics that lead to individuals (both children and adults) going missing, and determined that abuse and violence in the home are strong indicators. Knowing this, it is important to develop proper procedures to apply classification methods to data that detail child abuse and neglect. This topic was researched extensively by *Russell* and *Sledjeski et al.* Repeat missing persons are also a problem for justice workers; understanding why individuals go missing again helps build predictive models to indicate when an individual is likely to go missing again.

In (*Russell*, 2015), it was considered how predictive analytics might aid in making decisions about child protection and the precautions that need to be taken when interpreting the results of various methods. Child protection agencies have many questions that could be answered by predictive or classification models, such as why some families may be more at risk to experience maltreatment or on whom should the protection agencies focus. There are four well-established standards by which predictive models in child protection are judged: validity, equity, reliability, and usefulness (*D'andrade et al.*, 2008). *Russell* reported that predictive models have been used in many aspects of child protection, but primarily for assessing the risk of a family to have neglect or violence in the future. The paper did not mention models being used to directly impact missing children cases, but efforts to forward intentional predictive modeling and to ensure safe data analytics are being used.

Sledjeski et al. looked for a pattern-centric approach to predicting and classifying which families are at risk for recurrent maltreatment. Classification and Regression Trees (CARTs) were used to model this problem. The data are comprised of features that fall into one of five domains and a classifier that labels families as low, moderate, or high risk for recurrent maltreatment. Variable-centric statistical techniques have been relied on in previous efforts to classify recurrent maltreatment (*Sledjeski et al.*, 2008). A new insight to the problem was hoped to be achieved by using CARTs, a pattern-centric approach. *Sledjeski et al.* found that CARTs correctly identified "88% of recurrent cases (sensitivity) and 36% of the non-recurrent cases (specificity)", compared to the 37% sensitivity and 89% specificity obtained using logistic regression. CARTs do a better job at predicting recurrent maltreatment cases but worse at predicting the non-recurrent cases. This is why it is important to choose the machine learning technique that best suited the problem being explored.

Fyfe et al. reported on the processes and challenges surrounding a police investigation of a missing persons case. In many places, police officers receive an unmanageable amount of missing persons reports; the UK alone receives over 300,000 missing persons reports in a single year (*NPIA*, 2011). Teenagers make up the majority of missing persons reported. All cases have the potential of being a larger or more severe problem, but 80% of missing persons are found within 24 hours. Without a way to initially differentiate severe and not severe cases, all cases are dealt with in the same basic approach (*Fyfe et al.*, 2015). When the missing person is reported, the reporter is asked a series of questions in an attempt to determine how at risk the missing individual is to be exploited or harmed. A police officer makes the final judgment of how at risk the individual is. Further investigation and inquiries are made according to the risk level given to the case. The longer the case stays open and the higher the risk at which an individual is, the more officers and other resources are assigned to the case. Once the missing person is found, a routine interview is preformed to gain information on where the individual has been and what caused them to go missing. *Fyfe et al.* reported in detail about the aspects of a missing persons investigation, making it clear that there are many complex and intensive decisions made at many points by officers; decisions that could be aided using feature selection and machine learning methods.

2.5 Cognitive Impairment Pre-Diagnostics Test

Alzheimer's disease (AD) is a serious disease that is forcasted to quadruple in prevalence in the older population in the next 50 years (*Evans et al.*, 1989; *Brookmeyer et al.*, 1998, 2007). The key characteristics identified to indicate mild cognitive impairment (MCI) or the first step towards AD are: a change in cognition, impairment in a cognitive domain, remaining independent in daily life activities, and no significant impairment in social or occupational functioning (*Albert et al.*, 2011; *McKhann et al.*, 2011; *Sperling et al.*, 2011). To help aid in diagnosing AD, machine learning methods have been employed in the field. DT and SVMs have been successfully used to build classification models to aid in AD diagnosis (*Salas-Gonzalez et al.*, 2010; *Dana and Alashqur*, 2014). Further, a more advanced technique, random forest, has been used for feature selection and diagnosing AD (*Gray et al.*, 2013). Random forests use many layers of DTs to build a classification model.

Dana and Alashqur reported on the effectiveness of DTs for pre-diagnosing AD. Through their research, they demonstrated the effectiveness and simplicity of a DT applied to the problem. The data set used consisted of information on an individual's gender, age, genetic causes, brain injury, and vascular disease. *Dana and Alashqur* found that these were the main contributors that seem to cause AD. The data set consisted of 17 data entries. The DT method used was an entropy DT. The approach was modeled theoretically, going through how the DT would be built when pertaining to the data used. *Dana and Alashqur* reported that DTs look like a promising avenue to continue researching, but they did not report on accuracy because of the small sample size of their data set. They used the entirety of the data set to construct the DT, they did not allow for extensive testing of their model with a testing set of data, but they did provide a framework

for future researchers to build off of.

A computer-aided technique developed to improve the accuracy of diagnosing early Alzheimer-type dementia was developed by *Salas-Gonzalez et al.* using SVMs and classification trees (CTs). The data used in the study were single photon emission computed tomography (SPECT) images of normal control brains and brains known to have AD. The data set used by *Salas-Gonzalez et al.* contained 41 normal SPECT images and 38 SPECT images with AD. The features used to train the models were voxels. A voxel is a unit that measures a point in three-dimensional space associated with medical images. Voxels can contain a variety of values representing the opacity, colour, or scalar value associated with a given pixel in the SPECT image. Which voxels to be used was determined by a threshold value. The higher the threshold used, the more extreme valued the voxels were, and the fewer voxels that were selected per image. Multiple thresholds were tested for each model. The leave-one-out method was used to train and test the models; using all but a single image to train the models and attempting to classify it, repeated for each image in turn. The results show that SVM was able to obtain a higher accuracy when the threshold was low, but CTs were only able to obtain a higher accuracy when the threshold was high (*Salas-Gonzalez et al.*, 2010).

Using a random forest technique, *Gray et al.* developed a multi-step process to combine four sets of data that could contain indicators of early AD and a classification model. The four sets of data used were cerebrospinal fluid, magnetic resonance imaging, positron emission tomography, and genetic features. A random forest was first applied to each data set, and the results were used to derive the similarities between the data set. These similarities were then used for manifold learning. Manifold learning uses the measures of similarities the random forest classifier provides to determine the most important features for each data set (*Gray et al.*, 2013). *Gray et al.* were able to select subsets of features using a complex manifold learning method and random forests to successfully build a classifier for AD. Using this model, *Gray et al.* were able to classify AD patients with 89% accuracy.

2.5.1 CAMCI

The Computer Assessment of Mild Cognitive Impairment (CAMCI) was developed and tested by *Saxton et al.* The test is comprised of a series of small tasks the patient performs on a tablet that are scored and the data recorded. The sensitivity and specificity of the test were compared with a more widely used clinical examination, the Mini Mental State Examination (MMSE) (*Nasreddine et al.*, 2005). CAMCI was used to collect data on a group of individuals who were above the age of 60 and who had no previous diagnosis of MCI or AD. The final data set consisted of 524 participants. The CAMCI data were then used to build a CART using IBM SPSS software (*SPSS, IBM*, 2013). Using 10-fold cross validation, the final tree obtained 86% sensitivity and 94% specificity (*Saxton et al.*, 2009) These scores were much higher than the scores of MMSE, which achieves a sensitivity of 45% and a specificity of 80% (*Nasreddine et al.*, 2005).

The scores obtained by CAMCI were remarkably high and attracted attention from the AD community. *Ursenbach et al.* took particular interest in testing the reproducibility of the work by *Saxton et al.*. The

method described by *Saxton et al.* was reproduced in programming languages R (*R Core Team*, 2014) and python (*Pedregosa et al.*, 2011). The sensitivity and specificity obtained by both methods were drastically lower than the reported 86% sensitivity and 94% specificity. It was found that this difference was caused by *Saxton et al.* reporting on a DT that was over fit to their data. With close inspection of the method they used in SPSS, it was found that the final DT for the 10-fold cross validation method trained and tested with all the data as opposed to a DT trained with a training set of data and tested with a separate testing set of data. To support their theory, *Ursenbach et al.* produced a method in which the DT produced is trained and tested with all the data. The results obtained were much closer to the reported results, 92% sensitivity and 96% specificity for R and 89% sensitivity and 98% specificity for Python. Upon finding this error in *Saxton et al.* analysis and method, a new method was proposed. Using a Logistic Regression model, a sensitivity of 76% and a specificity of 72% were obtained (*Ursenbach et al.*, 2018). The conclusions warn readers to be cautious when applying machine learning methods and to fully understand what model outputs represent and can be interpreted to mean before reporting on results.

3 Machine Learning

Machine learning is a useful tool often used when dealing with large sets of data. It can extract meaningful information from data that can then be used to create a classification model. A classification model is used to determine the likely outcomes of unknown events in data (*Kotsiantis et al.*, 2007). This section introduces the underlying machine learning methods used in this study, namely the Naive Bayesian and DT classification methods. Further information and detail on the specified algorithms can be found in, e.g., (*Elkan*, 1997) and (*Quinlan*, 1993). The coding language and package used for the algorithms comes from the Scikit-learn library for python (*Thirion et al.*, 2016). Section 3.1 describes Naive Bayesian classification. Section 3.2 outlines two DTs. Section 3.3 describes the feature selection process and the two feature selection methods used.

3.1 Naive Bayesian Classification

Bayes' theorem describes the probability of an event occurring based on past knowledge of conditions (features) and whether they were true or false at the time of the event. The Naive Bayesian classifier is built off of this theorem with further assumptions of independence between the features using what is called the probability function. The probability function for Bayes' Theorem is

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)},$$

where x is the value of a predictor and c is a given class. $P(c|x)$ is the posterior probability of a class given a predictor and is calculated from the likelihood or probability of the predictor given the class, $P(x|c)$, the prior probability of the class, $P(c)$, and finally the prior probability of the predictor, $P(x)$. The Naive Bayesian classifier is simpler to build than some classification methods because it does not use complicated iterative classification methods and therefore works well for large data sets (*Sayad*, 2010–2017).

A detailed example is now given to provide a basic understanding of how the Naive Bayesian classification process works.

The raw data used in this example can be seen in Table 3.1. There are three features, "Crime Committed", "Criminal Friends", and "Grade 12 Education", that correspond to different outcomes of the classification feature "Re-offended" ("Yes" and "No").

For a set of features, X, the prediction value, X_{mk}, of the classification feature, X_m, for each class, k needs to be calculated. In this case, the classification feature is "Re-offend", and the classes are "Yes" or "No".

Crime Committed	Criminal Friends	Grade 12 Education	Re-offend
Robbery	No	No	Yes
Robbery	No	No	No
Robbery	No	No	Yes
Motor Vehicle Theft	No	No	No
Motor Vehicle Theft	No	Yes	Yes
Motor Vehicle Theft	Yes	Yes	No
Motor Vehicle Theft	Yes	Yes	Yes
Motor Vehicle Theft	Yes	No	No
Robbery	Yes	Yes	No
Robbery	No	Yes	Yes

Table 3.1: Re-offense Raw Data Table for Naive Bayesian Example

When there are multiple features, m_1, m_2, ..., $m_{|X|}$, the equation for the prediction value is,

$$X_{mk} = \text{argmax}_{x_j \in X} P(x_j) \prod_{i=1}^{|X|} P(m_i|x_j),$$

where $|X|$ is the number of elements in the data set X. It is applied to all classifications, X_{mk}, and the largest resulting value is the final classification, likely or not likely to re-offend. The $P(m_i|x_j)$ is given by

$$P(m_i|x_j) = \frac{n_c + wp}{n + w},$$

where n is the number of data matching the desired classification value x_j, n_c is the number of data that match both the desired classification value x_j and the feature value m_i, p is the prior estimate of the value of $P(m_i|x_j)$, and w is called the equivalent sample size[1] and adds a weighted respective of the observed data to the value p. For a deeper explanation of this formula see (*Kubat*, 2015). In this example, the value of w is chosen to be 3 for all calculations because there are three features other than the classification feature in the data set, and the p value is 0.5 to reflect that the values of each feature in the data set are binary (*Meisner*, 2003).

From the raw data in Table 3.1, a frequency table is calculated that details the number of times an instance has occurred when certain conditions are present; see Table 3.2. The Naive Bayesian classifier assumes the instances and conditions are independent of each other; this is the naive aspect of the classifier.

To give a full classification or prediction, the X_{mk} value must be calculated for all classification outcomes given the predictor, and the resulting outcome is the highest probability value. For this example, the goal is to determine if an individual is likely to re-offend or not when they have committed a robbery, have criminal friends, and do not have a grade 12 education.

[1]This value is also referred to as the m-estimate, not to be confused with the variable m that represents a feature in the equation.

Frequency Table		Re-offended	
		Yes	No
Crime Committed	Robbery	3	2
	Motor Vehicle Theft	2	3
Criminal Friends	Yes	1	3
	No	4	2
Grade 12 Education	Yes	3	2
	No	2	3

Table 3.2: Frequency Table

The probability, $P(m_i|x_j)$, for the outcome or class "Yes" and the attributes or predictors "Robbery", "Criminal Friends$_{Yes}$", and "Grade 12$_{No}$" is calculated in equation (3.1) using the values in the frequency table. Lastly, the X_{mk} value is calculated using the calculated probabilities.

$$
\begin{aligned}
P(\text{Robbery}|\text{Yes}) &= \frac{3+3*0.5}{5+3} = 0.56 \\
P(\text{Criminal Friends}_{Yes}|\text{Yes}) &= \frac{1+3*0.5}{5+3} = 0.31 \\
P(\text{Grade } 12_{No}|\text{Yes}) &= \frac{2+3*0.5}{5+3} = 0.43 \\
P(\text{Robbery}) &= 5/10 = 0.5 \\
P(c) &= P(\text{Yes}) = 5/10 = 0.5 \\
X_{mk} &= P(\text{Robbery}|\text{Yes}) * P(\text{Criminal Friends}_{Yes}|\text{Yes}) * P(\text{Grade } 12_{No}|\text{Yes}) * P(\text{Yes}) \\
&= 0.56 * 0.31 * 0.43 * 0.5 = 0.037
\end{aligned}
\tag{3.1}
$$

The results for the first X_{mk}, $P(\text{Yes}|\text{Robbery})$, are calculated, the next value to be calculated determines the probability the individual would *not* re-offend given the previous conditions. We obtain the needed

probability values to determine the X_{mk} value for $P(\text{No}|\text{Robbery})$,

$$\begin{aligned}
P(\text{Robbery}|\text{No}) &= \frac{2+3*.5}{5+3} = 0.43 \\
P(\text{Criminal Friends}_{Yes}|\text{No}) &= \frac{3+3*.5}{5+3} = 0.56 \\
P(\text{Grade 12}_{No}|\text{No}) &= \frac{2+3*.5}{5+3} = 0.56 \\
P(c) &= P(\text{No}) = 5/10 = 0.5 \\
X_{mk} &= P(\text{Robbery}|\text{No}) * P(\text{Criminal Friends}_{Yes}|\text{No}) * P(\text{Grade 12}_{No}|\text{No}) * P(\text{No}) \\
&= 0.43 * 0.56 * 0.56 * 0.5 = 0.069
\end{aligned} \tag{3.2}$$

The two results are then compared, and the higher value has the greater probability of happening, ultimately classifying the data as re-offending or not. The posterior probability for *not* re-offending is 0.069 and for re-offending after committing a robbery is 0.037. The posterior probability for *not* re-offending after committing a robbery is greater; therefore the Naive Bayesian classifier classifies an individual as not likely to re-offend if they have committed a robbery, have criminal friends, and do not have a grade 12 education. This classification process is more complicated as the number of features in the data set increases, but it works well for large amounts of data.

3.2 Decision Trees

DTs are a supervised machine learning method that classify data using a tree-like structure. DTs fall into the category of a supervised machine learning method because the training data used to build the model contain the classification feature (*Marsland*, 2015). DTs learn simple decision rules from the training data and structure these rules to form the tree-like model. The DT model can be used to classify new sets of data.

Figure 3.1 is a visual representation of the DT method. A DT is a classifier that contains nodes that are directly connected to the root node at the top of the tree by edges. Each node with an outgoing edge is called a test node; these nodes represent features in the data set. Nodes with no outgoing edges are called leaves; these nodes give a classification result. After the tree is built, it can be used to classify outcomes from a given data set.

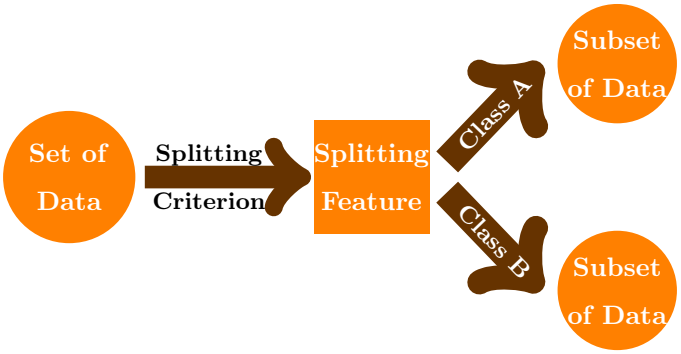

Figure 3.1: Decision Tree Classification

To build a DT, a set of training data must be chosen. DTs are built top-down in an iterative fashion. In each iteration, a variable is chosen from the training data set that best splits the remaining features; this variable is called the *splitting feature*. What constitutes the "best" splitting feature can be measured in different ways with different splitting criteria. In this thesis, the two splitting criteria used were entropy and Gini impurity. Entropy and Gini impurity are different measures of information gain (*Quinlan*, 1986). The splitting criterion determines the order in which the features appear in the decision tree, therefore having a large impact on the final structure of the model. At each iteration, the data are split down the different paths or outcomes according to the chosen feature, resulting in two subsets of the data. The resulting subsets of each iteration are used as the input for the next iteration. This is done until the subset of data results in a unified classification of the data, meaning there is only one type of prediction present in the classification feature. Figure 3.2 is an example of what a piece of a decision tree can look like.

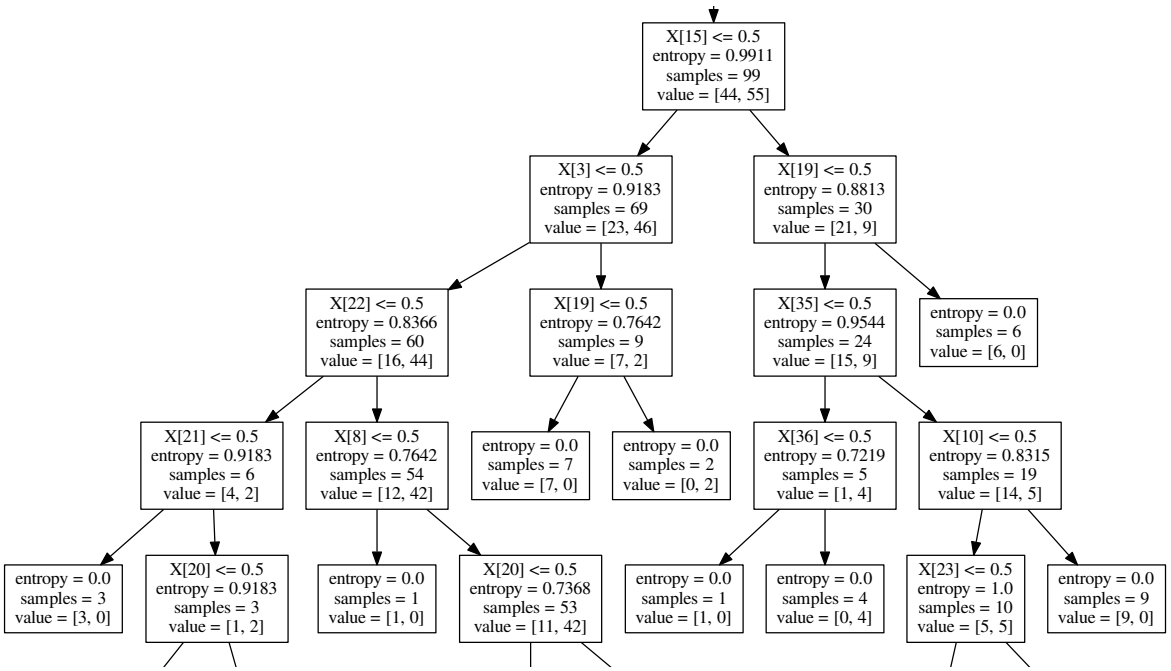

Figure 3.2: A DT Example

The following two subsections describe the entropy and Gini impurity splitting criteria. Further details on the methods and mathematical formulae discussed in these subsections can be found in the Scikit-learn python library (*Thirion et al.*, 2016).

3.2.1 Splitting Criteria

3.2.1.1 Entropy

Entropy DTs use the entropy measure to determine the best splitting feature when building the model. Entropy is the measure of information gain or loss (*Raileanu and Stoffel*, 2004). For a set of features, X, the entropy equation for a given feature, X_m, is

$$S(X_m) = \sum_{k=1}^{n_c} -p_{mk} \log_b p_{mk}, \tag{3.3}$$

where p_{mk} is the probability of a class k of feature m, n_c is the number of classes of that feature, and b is the base of the logarithm. The value of b is commonly one of three numbers, 2, Euler's number e, or 10. In this problem, the classification value is binary, and because of this, b is 2. If the number of class labels present in the data is equally distributed, the entropy equation yields 1. When the number of class labels is homogeneous, the entropy equation yields 0. In DTs, the entropy measurement is taken before and after splitting at a specific feature, and the difference is examined. The difference is the information gain. After calculating the information gain for all of the features, the optimal splitting feature is chosen, i.e., the feature that produces the largest information gain value.

To give a deeper understanding of how entropy is used and calculated, an example is given. The goal is to build a DT to predict whether an individual will re-offend given a set of features, in this case relating to the crime they committed. A set of raw data containing the classification feature, *Re-offended*, and a single feature, *Crime Committed*, is given in Table 3.3.

When deciding the splitting feature, two types of entropy values must be calculated. The first is the entropy of the classification feature, X_m, *Re-offended*. The calculations in equation (3.4) uses equation (3.3) and Table 3.4 to calculate the Entropy of the target feature, *Re-offend*.

$$\begin{aligned} S(\text{Re-offend}) &= \sum_{k=1}^{2} -P(\text{Re-offend}_k) \log_2 P(\text{Re-offend}_k) \\ &= -(P(\text{Re-offend}_{no}) \log_2 P(\text{Re-offend}_{no})) - (P(\text{Re-offend}_{yes}) \log_2 P(\text{Re-offend}_{yes})) \\ &= -(5/14 \log_2 5/14) - (9/14 \log_2 9/14) \\ &= 0.94 \end{aligned} \tag{3.4}$$

The next step is to calculate the entropy values for the model if it was to split on each feature and compare it to the entropy value for the feature *Re-offended*. In this example, there is only one feature to consider,

Crime Committed	Re-offend
Robbery	Yes
Robbery	Yes
Robbery	Yes
Robbery	No
Robbery	No
Motor Vehicle Theft	Yes
Motor Vehicle Theft	Yes
Motor Vehicle Theft	Yes
Motor Vehicle Theft	Yes
Aggravated Assault	Yes
Aggravated Assault	Yes
Aggravated Assault	No
Aggravated Assault	No
Aggravated Assault	No

Table 3.3: Raw Data

Re-offend		
Yes	No	Total
9	5	14

Table 3.4: Target Feature

Crime Committed. The equation to calculate the entropy for the model after splitting on a feature, F, and still considering the classifications of the target feature, X_m, is

$$S_{split}(X_m, F) = \sum_{k=1}^{n_c} p_k S(F_{mk}), \tag{3.5}$$

where n_c is the number of classes in feature F, p_k is the probability of class k of feature F and $S(F_{mk})$ is the entropy of the model after splitting on feature F while still taking the classification feature, X_m, into consideration. Considering *Re-offended* as X_m and *Crime Committed* (C.C. in equation (3.6)) as F, the calculations can be seen in equation (3.6), using equation (3.5) and the frequency table in Figure 3.5.

Likelihood Table		Re-offend		
		Yes	No	
Crime Committed	Robbery	3	2	5/14
	Motor Vehicle Theft	4	0	4/14
	Aggravated Assault	2	3	5/14
		9/14	5/14	

Table 3.5: Frequency Table

$$
\begin{aligned}
S_{split}(\text{Re}-\text{offend}_\text{k}, \text{C.C.}) &= P(\text{Robbery})S(\text{Robbery}_\text{k}) \\
&\quad + P(\text{MotorVehicleTheft})S(\text{MotorVehicleTheft}_\text{k}) \\
&\quad + P(\text{AggravatedAssault})S(\text{AggravatedAssault}_\text{k}) \\
&= 5/14 * (\sum_{k=1}^{n_c=2} -P(\text{Robbery}_k)\log_2 P(\text{Robbery}_k)) \\
&\quad + 4/14 * (\sum_{k=1}^{n_c=2} -P(\text{MotorVehicleTheft}_k)\log_2 P(\text{MotorVehicleTheft}_k)) \\
&\quad + 5/14 * (\sum_{k=1}^{n_c=2} -P(\text{AggravatedAssault}_k)\log_2 P(\text{AggravatedAssault}_k)) \\
&= 5/14 * [(-P(\text{Robbery}_{yes})\log_2 P(\text{Robbery}_{yes})) \\
&\quad + (-P(\text{Robbery}_{no})\log_2 P(\text{Robbery}_{no}))] \\
&\quad + 4/14 * [(-P(\text{MotorVehicleTheft}_{yes})\log_2 P(\text{MotorVehicleTheft}_{yes})) \\
&\qquad + (-P(\text{MotorVehicleTheft}_{no})\log_2 P(\text{MotorVehicleTheft}_{no}))] \\
&\quad + 5/14 * [(-P(\text{AggravatedAssault}_{yes})\log_2 P(\text{AggravatedAssault}_{yes})) \\
&\qquad + (-P(\text{AggravatedAssault}_{no})\log_2 P(\text{AggravatedAssault}_{no}))] \\
&= 5/14 * [(-P(3/5)\log_2 P(3/5)) + (-P(2/5)\log_2 P(2/5))] \\
&\quad + 4/14 * [(-P(4/4)\log_2 P(4/4)) + (-P(0/4)\log_2 P(0/4))] \\
&\quad + 5/14 * [(-P(2/5)\log_2 P(2/5)) + (-P(3/5)\log_2 P(3/5))] \\
&= 5/14 * 0.971 + 4/14 * 0.0 + 5/14 * 0.971 \\
&= 0.693.
\end{aligned} \tag{3.6}
$$

With equations (3.4) and (3.6), the information gain can be calculated as

$$Gain(X_m, F) = S(X_m) - S_{split}(X_m, F),$$

where $S(X_m)$ is the entropy of the classification feature, in this example *Re-offended*, and $S_{split}(X_m, F)$ is the entropy calculation for the observed feature after splitting, in this example *Crime committed*. The information gain is calculated for each feature, and the feature with the greatest information gain is chosen as the splitting feature. The greater the information gain is, the more homogeneously grouped the values of the classification feature are after splitting on the splitting feature. If there were more features, the process is repeated for each of the features, and the one with the largest resulting information gain is chosen as the splitting feature.

3.2.1.2 Gini Impurity

A Gini DT uses the Gini impurity measure as its rule to determine the best splitting feature when creating a DT model from data. Gini impurity measures the likelihood of a random sample being classified correctly if the classification is randomly selected according to a branch distribution (*Quinlan*, 1986). For a set of features, X, the Gini impurity equation for a given classification feature, X_m, is

$$\begin{aligned} G(X_m) &= \sum_{k=1}^{n_c} p_{mk}(1 - p_{mk}) \\ &= 1 - \sum_{k=1}^{n_c} {p_{mk}}^2 \end{aligned} \tag{3.7}$$

where p_{mk} is the probability of class k of feature m, and n_c is the number of classes of the classification feature.

The Gini gain is calculated for each feature. The Gini gain measures the difference in values of the Gini impurity of the classification feature and after splitting on a feature. The feature with the largest Gini gain is selected to be the splitting feature and the process repeats. In other words, Gini impurity tries to select the feature that minimizes the probability of misclassification. Below is a detailed example of how a Gini DT is built.

The data used to build the Gini DT in this example are given in Table 3.6. The classification feature that the DT is built to predict is R, the different data entries are labeled in column X in the form x_n, and the features are *M, N,* and *Q*.

X	M	N	Q	R
x_1	1	3	2	2
x_2	2	3	2	1
x_3	2	2	1	1
x_4	1	2	1	2
x_5	2	1	3	2
x_6	1	1	3	2
x_7	2	3	1	1
x_8	1	2	2	2

Table 3.6: Raw Data for Gini Example

The initial step is to calculate the probability of each class in the classification feature R. This is done by summing all instances of the same class and dividing that by the total number of data entries, as in

equation (3.8):

$$\begin{aligned} P(R_1) &= 3/8, \\ P(R_2) &= 5/8. \end{aligned} \tag{3.8}$$

After the probability of the classification feature R is calculated, it is used to obtain the Gini value for the entire system using equation (3.7):

$$\begin{aligned} G(R) &= \sum_{k=1}^{2} [P(R_k)(1 - P(R_k))] \\ &= [3/8(1 - 3/8)] + [5/2(1 - 5/8)] \\ &= 15/32. \end{aligned} \tag{3.9}$$

The next step is to calculate the Gini value for the system as if it split on each feature. These values are used to calculate the Gini gain, the final value used to determine the splitting feature of the DT. To do this, Equation (3.7) is applied to all features and their possible labels or values. These values are then used to calculate the Gini impurity value of the model as if it split on one of the features. For a set of features, X, the Gini impurity equation to determine the Gini value after splitting on a given feature, F, is

$$G_{split}(X_m, F) = \sum_{k=1}^{n_c} \frac{n_k}{n} G(F_{mk}) \tag{3.10}$$

where n_c is the total number of classes for that feature, n is the total number of data points, n_k is the number of data points of class k, and $G(F_{mk})$ is the Gini value of the model after splitting on feature F while still taking the classification feature X_m into consideration.

Looking at the set of calculations in equation (3.11), the Gini value when splitting on feature M is calculated using the values found in Table 3.7 and equations (3.7) and (3.10):

Frequency Table	R	
	$R = 1$	$R = 2$
$M = 1$	0	4
$M = 2$	3	1

Table 3.7: Frequency Table for M

$$
\begin{aligned}
G(M_1) &= 1 - \sum_{k=1}^{n_c=2} (P(M_{1k}))^2 \\
&= 1 - \left[\left(\frac{0}{4} \right)^2 + \left(\frac{4}{4} \right)^2 \right] \\
&= 0, \\
G(M_2) &= 1 - \sum_{k=1}^{n_c=2} (P(M_{2k}))^2 \\
&= 1 - \left[\left(\frac{3}{4} \right)^2 + \left(\frac{1}{4} \right)^2 \right] \\
&= \frac{3}{8}, \\
G_{split}(R, M) &= \sum_{k=1}^{n_c=2} \frac{n_k}{n=8} G(M_k) \\
&= \frac{4}{8}(0) + \frac{4}{8} \left(\frac{3}{8} \right) \\
&= \frac{3}{16}.
\end{aligned}
\tag{3.11}
$$

The Gini calculations for features N and Q follow the same logic. The calculations and results for each are seen in equations (3.12) and (3.13) respectively.

Frequency Table	R	
	$R = 1$	$R = 2$
$N = 1$	0	2
$N = 2$	1	2
$N = 3$	2	1

Table 3.8: Frequency Table for N

$$
\begin{aligned}
G(N_1) &= 1 - \sum_{k=1}^{n_c=2} (P(N_{1k}))^2 \\
&= 1 - \left[\left(\frac{0}{2} \right)^2 + \left(\frac{2}{2} \right)^2 \right] \\
&= 0 \\
G(N_2) &= 1 - \sum_{k=1}^{n_c=2} (P(N_{2k}))^2 \\
&= 1 - \left[\left(\frac{1}{3} \right)^2 + \left(\frac{2}{3} \right)^2 \right] \\
&= \frac{4}{9} \\
G(N_3) &= 1 - \sum_{k=1}^{n_c=2} (P(N_{3k}))^2 \\
&= 1 - \left[\left(\frac{2}{3} \right)^2 + \left(\frac{1}{3} \right)^2 \right] \\
&= \frac{4}{9} \\
G_{split}(R, N) &= \frac{2}{8}(0) + \frac{3}{8} \left(\frac{4}{9} \right) + \frac{3}{8} \left(\frac{4}{9} \right) \\
&= \frac{1}{3}
\end{aligned}
\tag{3.12}
$$

Frequency Table	R	
	$R = 1$	$R = 2$
$Q = 1$	2	1
$Q = 2$	1	2
$Q = 3$	0	2

Table 3.9: Frequency Table for Q

$$
\begin{aligned}
G(Q_1) &= 1 - \sum_{k=1}^{n_c=2} (P(Q_{1k}))^2 \\
&= 1 - \left[\left(\frac{2}{3}\right)^2 + \left(\frac{1}{3}\right)^2 \right] \\
&= \frac{4}{9} \\
G(Q_2) &= 1 - \sum_{k=1}^{n_c=2} (P(Q_{2k}))^2 \\
&= 1 - \left[\left(\frac{1}{3}\right)^2 + \left(\frac{2}{3}\right)^2 \right] \\
&= \frac{4}{9} \\
G(Q_3) &= 1 - \sum_{k=1}^{n_c=2} (P(Q_{3k}))^2 \\
&= 1 - \left[\left(\frac{0}{2}\right)^2 + \left(\frac{2}{2}\right)^2 \right] \\
&= 0 \\
G_{split}(Q) &= \frac{3}{8}\left(\frac{4}{9}\right) + \frac{3}{8}\left(\frac{4}{9}\right) + \frac{2}{8}(0) \\
&= \frac{1}{3}
\end{aligned}
\tag{3.13}
$$

Having calculated the Gini value for each feature, these values are used to calculate the Gini gain,

$$
Gain(x_m, F) = G(X_m) - G_{split}(X_m, F), \tag{3.14}
$$

where $G(X_m)$ is the Gini value of the classification feature being analyzed and $G_{split}(X_m, F)$ is the Gini value after splitting on feature F.

Using equation (3.14), the Gini gain is calculated for all features in equation (3.15).

$$
\begin{aligned}
Gain(R, M) &= G(R) - G_{split}(R, M) \\
&= \frac{15}{32} - \frac{3}{16} = 0.28125 \\
Gain(R, N) &= G(R) - G_{split}(R, N) \\
&= \frac{15}{32} - \frac{1}{3} = 0.13542 \\
Gain(R, Q) &= G(R) - G_{split}(R, Q) \\
&= \frac{15}{32} - \frac{1}{3} = 0.13542
\end{aligned}
\tag{3.15}
$$

Using the values calculated in (3.15), the feature that has the greatest Gini gain, which is the feature that minimizes the misclassification rate, is selected as the splitting feature; in this case feature M. This feature

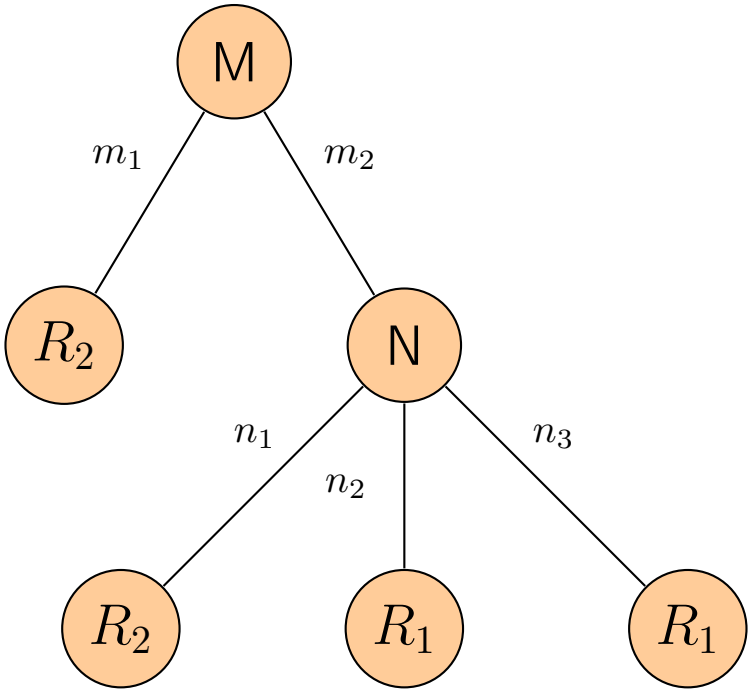

Figure 3.3: DT Model Produced from Gini DT Method

is then put at the top of the tree as the root node, and the data are split accordingly. Figure 3.3 shows how this process splits up the data. All values that are split down branch m_1 result in a classification of 2, resulting in that node producing a classification value. Following the other branch, m_2, a set of data with various classification values is left. The splitting process for the Gini splitting criterion is repeated on this subset of the data, calculating the next splitting feature. For this set of data, both feature N and Q obtain the same Gini gain values, and either can be chosen as the next splitting feature; in this case N was chosen. The steps are repeated until only data with a single classification feature value are left in each branch.

3.3 Feature Selection

Feature selection is the process of reducing the number of features under consideration in a model (*Mallick*, 2015). The goal is to capture the initial data set used to build the model with a new, smaller set of features. The selected set is used to build a model that should retain the predictive capabilities of a model built with the initial data. Figure 3.4 illustrates how the feature selection process is conducted. Starting with the full set of raw data, features are eliminated. After elimination, a model is engineered and validated. If the validation step is successful, the relevant data have been obtained, and if it is not successful, the elimination, engineering, and validation steps are repeated. The following sections outline the SBC and the SDTC.

Figure 3.4: Feature Selection

3.3.1 Selective Bayesian Classifier

An SBC is a method of feature selection that combines DTs and the Naive Bayesian classifier (*Ratanamahatana and Gunopulos*, 2002). It improves upon the Naive Bayesian classifier by removing redundant or less important features from the data set. Recalling that DTs create a tree structure of features, these features can be categorized by the level at which they fall on the tree, with the root or starting node of the tree being level one. The DT algorithm does not have an overly complicated structure, but even so, it can become quite deep (have many layers) for data sets with many features. The levels get more complex and have less information gain the deeper into the tree they get. This is drawn from the fact that DTs split from the best possible node as early as possible, the nodes that are the "most" important (*Ratanamahatana and Gunopulos*, 2002). The idea is that one can use the top-most layers of the DT to select the most relevant features. These features are then the only ones used when training and testing the Naive Bayesian classifier. As described in (*Ratanamahatana and Gunopulos*, 2002), the steps to the Selective Bayesian classifier can be seen in Figure 3.5.

There were shortcomings to this method and aspects that could be generalized. The SBC uses specific values, such as using the top three layers seen in Step 3, the specific values were not validated but suspected to be appropriate because they performed acceptably in testing. Also, only one type of DT was tested, an entropy-based DT. Testing other types of DT should help to validate the approach and algorithm. Lastly, the final model built was Naive Bayesian, which does work well overall for most sets of data, but does not scale well in terms of accuracy with larger sets of data when compared to the scalability of DT models (*Kohavi*, 1996). For these reasons, the Selective Bayesian classification is not always optimal, and the alternative version, SDTC, was proposed and implemented.

1. Shuffle the training data and take a 10% sample.

2. Run Decision tree on data from step 1.

3. Select a set of attributes that appear only in the first 3 levels of the decision tree as relevant features.

4. Repeat 5 times (step 1-3)

5. Form a union of all attributes from the 5 rounds.

6. Run Naive Bayesian classifier on the training and test data using only the final features selected in step 5.

Figure 3.5: Selective Bayesian Classifier (*Ratanamahatana and Gunopulos*, 2002)

3.4 Selective Decision Tree Classifier

SDTC is a classification method based on feature selection. SDTC follows the same logic as SBC, but it is generalized to allow for a wider use. Also, the DT method is used to build the final model instead of performing Naive Bayesian classification on the reduced feature set. SDTC first selects the features using a tree hierarchy and then uses those selected features to build a DT model. The steps to the SDTC can be seen in Algorithm 1.

Algorithm 1 SDT

1: **procedure** SELECTIVE DECISION TREE CLASSIFICATION (data, N=number of trees, M=depth of feature selection)
2: Split the data into training and testing sets.
3: Shuffle the training data and split into N sets.
4: **for** $i = 1$ to N training sets **do**
5: Build a decision tree with training set i.
6: **for** $j = 1$ to M layers in DT **do**
7: Place features on level j into the selected feature set.
8: Using the selected features of the training dataset (the union of all features on M levels of N DTs), build a Decision Tree Model.
9: Using only the selected features of the testing dataset, test the model.

The initial step is to split the data into a training and testing set such that the data used to build the

model are not used to test the model. When data used to build the model are used to test the model, a deceivingly high accuracy can be obtained (*Domingos*, 2012), a phenomenon known as *overfitting*. Such models are potentially dangerous because their classification capabilities on new data are often diminished. To ensure the training data set has an equivalent representation of classification values, a random stratified sampling technique is used (*Lumley et al.*, 2004). This technique ensures the weighting of the classification values in the training data set give a good representation of the full data set.

After this step, the training set is split into N pieces. When choosing the value of N, the size of the data set should be considered. The smaller the data set, the smaller N should be. If N was the size of half the amount of data in the data set being used, each split would only contain at most two data points, an unreasonable amount of data to train a machine learning model. With the first N-split of the data, a decision tree is built. The features selected are the ones that appear in the top M layers of the ensuing DT. The value of M chosen for SDTC ranges, generally, from 1, the root of the tree, to 5. For example, if M is equal to 2, the top two layers would be selected from the tree seen in Figure 3.6, selecting the features labeled $X[15]$, $X[3]$, and $X[19]$. Any depth level up to the final base layer of the tree can be chosen as M, but the more layers that are included, the larger amount of features selected. This process is repeated N times, once for each piece of the training set. A union of the selected features is taken from the decision trees to make up the final selected feature set.

The final DT model is built with the training data using only the selected features. The model is then tested with only the designated selected features present in the testing data set. This process is repeated, varying the depth of selection (M in the algorithm) until an appropriate combination of number of features and accuracy scores is obtained. The final combination of features can vary depending on the splitting criterion used or the details of the analysis.

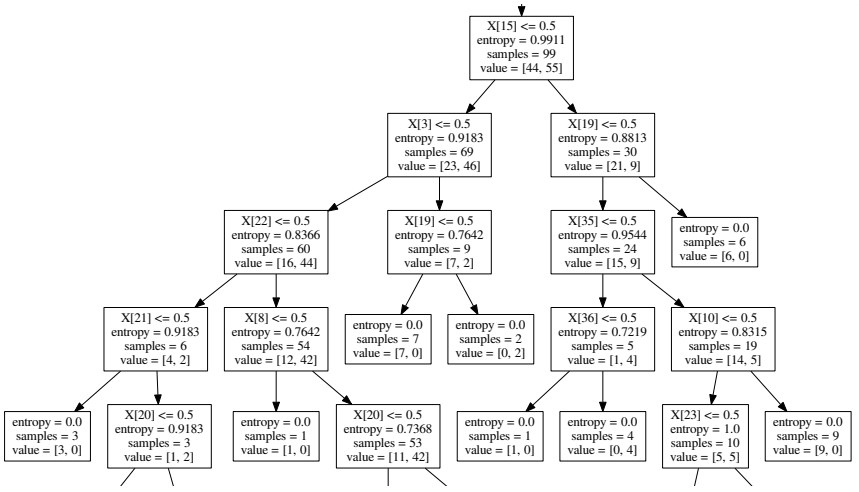

Figure 3.6: A DT Example

4 Experimental Design

Below are the research objectives of the thesis:

Objective 1: The first objective was to implement the SDTC method and test it using various data sets. The method was implemented using the Python programming language and various packages. The SDTC was then tested using the three different data sets described in Section 4.2, Section 4.3, and Section 4.4.

Objective 2: The second objective was to compare the specific implementation and capabilities of the SDTC to the SBC. This was done to compare the strengths and weaknesses of both feature selection and classification methods.

Objective 3: The third objective was to demonstrate an application of SDTCs by comparing the features selected to a test that has already been shortened using a different method. This was done by applying SDTCs to the LSI data and comparing the selected features with the LSI-R:SV features. The features in the LSI-R:SV were selected by a group of psychologists and criminologists. If the scores and features are similar between the models built with LSI-R:SV features and the selected feature sets, it validates LSI-R:SV; if the scores of the selected feature set models are greater, it gives practitioners an empirically validated alternative to LSI-R:SV.

4.1 Experimental Method

4.1.1 SDTC

The SDTC was performed on each data set. The data were first split into 80% and 20% for training and testing respectively. The training data were split into five random pieces. When splitting the training data into five pieces, a standardized random seed value was used to shuffle the data. Each distinct value of a random seed results in a different yet reproducible random shuffle of the data. Using the split data, five DTs were built. Various depth levels used to select the features from the layers of the DTs were tested and reported on. The depth level selected corresponds to the levels of the DT that were used to determine the selected feature set. These selected features were then used to build a final DT classifier that was tested with the testing data. Different splitting criteria could hierarchically place the features of a data set in different patterns. To test whether or not the splitting criteria of the DT affect the accuracy of the overall model, the process was repeated for each of the Gini DTs and entropy DTs. The features selected and relevant validity scores are reported.

4.1.2 SBC

The SBC was performed on each of the data sets. The data were first split into 80% and 20% for training and testing respectively. Five Entropy DTs were built using a random 10% sample with replacement of the training data as stated in the steps found in Figure 3.5. For each DT, the top 3 layers of features were added to the selected feature set. These selected features were then used to build a final Naive Bayesian classifier that was tested with the testing data. The features selected and relevant validity scores are reported.

4.1.3 Validity Scores

The validity scores used to compare models in this section are the statistics calculated from the true negatives, false negatives, false positives, and true positives of the applicable model. The values in the various tables are calculated as follows,

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN},$$

$$\text{Sensitivity} = \frac{TP}{TP + FN},$$

$$\text{Specificity} = \frac{TN}{TN + FP},$$

$$\text{Positive Predictive Value (PPV)} = \frac{TP}{TP + FP},$$

$$\text{Negative Predictive Value (NPV)} = \frac{FP}{FP + TN},$$

$$\text{False Negative Rate (FNR)} = \frac{FN}{TP + FN},$$

$$\text{False Discovery Rate (FDR)} = \frac{FP}{TP + FP},$$

where FP is the number of false positives, TP is the number of true positives, FN is the number of false negatives, and TN is the number of true negatives. These values come from a confusion matrix. Figure 4.1 shows the relationship a confusion matrix has to the TP, FP, TN, and FN in terms of a binary classifier.

Accuracy measures how many times a model correctly predicts the classification feature, true or false, over the total number of predictions. Sensitivity measures how many true values in the data for which the model produced the value true. Specificity measures how many false values in the data for which the model predicted the value false. The false negative rate = 1− sensitivity and measures how often the model prediction produced the value false, but the classification feature was true. The false positive rate = 1− specificity and measures how often the model prediction produced the value true, but the classification feature was false. An ROC curve visually represents the sensitivity and specificity of a model. The sensitivity is plotted against the specificity for models built with various cut off points for a given parameter. The ROC curves are not reported on in this thesis, but are available upon request.

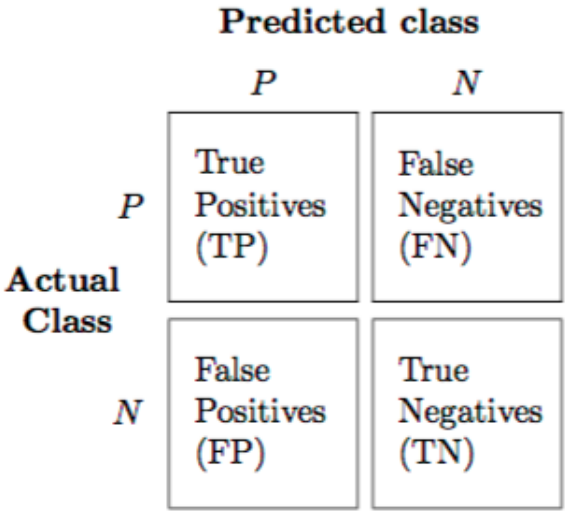

Figure 4.1: Confusion Matrix (*Townsend*, 1971)

4.2 LSI Experiments

The LSI data set used is made up of over 70,000 entries, each consisting of 43 features and whether or not the individual recidivated. The data set being used was provided by Ontario Ministry of Community Safety and Correctional Services (MCSCS). The data were collected by correctional staff during 2010–2011 from offenders who had been given a sentence from three months to two years. An offender with a sentence greater than two years is transferred to the federal correctional authority and is not administered the tool by MCSCS because such individuals are not under provincial jurisdiction. The forty-three features are split into eight subgroups: Criminal History (8 features), Education/Employment (8 features), Family/Marital (4 features), Leisure/Recreation (2 features), Companions (4 features), Substance Abuse (8 features), Pro-Criminal Attitudes (4 features), and Antisocial Pattern (4 features) (*Andrews and Bonta*, 2010). The individual features are not weighted, but a weight is implicitly applied to different subgroups of features by the number of features present in each subgroup.

The three largest subgroups of features in the LSI test are Criminal History, Education/Employment, and Substance Abuse. These are the categories that correctional professionals have deemed most important for assessing the risk at which an individual is to recidivate by making them the largest or most heavily weighted subgroups.

Figure 4.2 is the histogram for individuals who did not recidivate, 0, and who did recidivate, 1. Figure 4.3 is the histogram for the overall total score of each data instance. There is a clear skew towards the lower scores. The histograms for all of the individual features within the data set can be found in Appendix A.

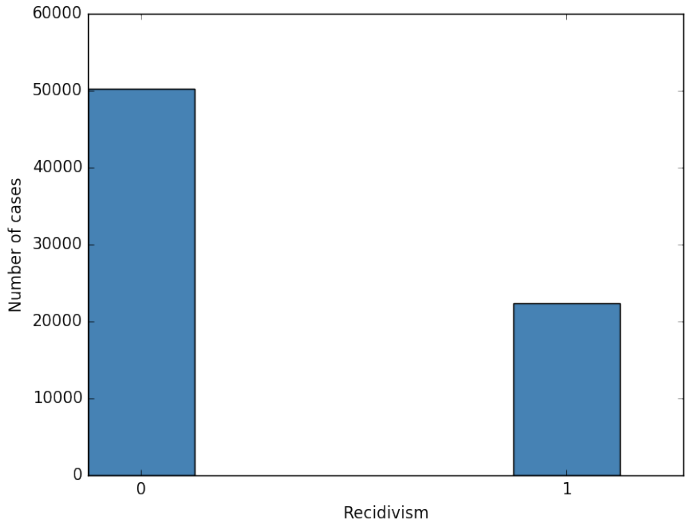

Figure 4.2: Re-offence Distribution

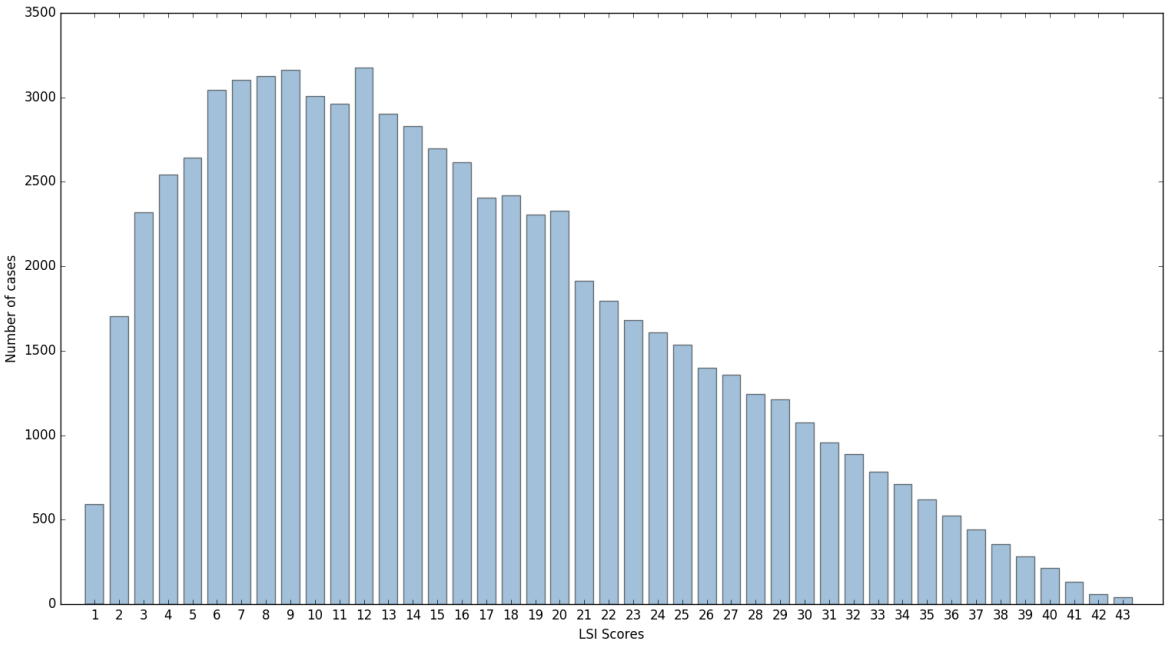

Figure 4.3: Total Score Distribution

The goal by building a model on the given data is to be able to provide correctional professionals with a model that can aid in the prediction of re-offenders. It is also beneficial apply feature selection to the data to provide correctional professional with a list of features that are more important in predicting re-offenses. The correctional professionals were looking for a model that uses less features but retains the accuracy of a

model built with all the features.

4.2.1 Results

The data were split up into two groups, 80% training and 20% testing using stratified sampling according to recidivism. This put 58,182 data points in the training set and 14,545 in the testing set. The results were obtained by applying the SDTC technique on these split data sets.

Table 4.1 gives a summary of the results. It is organized to display from which specific depth levels the features were selected, the number of features selected for each splitting criterion used, and the accuracy (%). At each depth level, the features selected are those from the current and all previous levels. The row labeled "All Features" is the result from an entropy DT built with all the features in the data set. The row labeled "SBC" is the results from an SBC applied to the data. The highlighted green columns are the best accuracy obtained for each splitting criterion.

Depth	Splitting	# Features Selected	Acc. (%)
1	Gini	1	76.1
	Entropy	1	76.1
2	Gini	5	72.4
	Entropy	5	72.4
3	Gini	11	73.0
	Entropy	11	73.0
4	Gini	26	72.2
	Entropy	26	72.5
All Features	Entropy	43	71.3
SBC	Entropy	15	73.9

Table 4.1: SDTC Results for LSI Data

At a depth level of 1, both a Gini SDTC and an entropy SDTC selected only feature *A18* as the most relevant feature. At a depth level of 2, a Gini and an entropy SDTC selected the features *A11*, *A18*, *A210*, *A423*, and *A843*.

Table 4.2 is a summary of the features that the SDTC selected using a depth level of 3 and the features selected by the SBC. Each method is given a distinct colour; if a feature is selected by that method, the box is filled in with the designated colour. Row *SDTC* represents both the Gini and entropy splitting criterion results; the same features were selected by both when using a depth level of 3.

	A11	A12	A13	A14	A16	A17	A18	A29	A210	A216	A217	A423	A524	A525	A733	A735	A737	A843
SDTC																		
SBC																		

Table 4.2: The Overlay of Features Selected at Depth 3

Table 4.3 compares the validity scores for the SDTC with the highest accuracy score from Table 4.1 compared to an entropy DT built with all the features in the data and an SBC applied to the data.

	Entropy & Gini	All Features Entropy	SBC
Accuracy	76.1	71.3	73.9
Sensitivity	33.7	24.5	18.0
Specificity	95.0	92.2	98.8
PPV	75.1	58.5	87.0
NPV	76.2	73.2	73.0
FPR	4.98	7.7	1.19
FNR	66.2	75.4	81.9
FDR	24.8	41.4	12.9

Table 4.3: Statistics for the Best Selected Model and the Original Full Model in Percent (%)

4.2.2 Discussion

Looking at Table 4.1, both the Gini and entropy splitting criteria performed best at the first depth level. The single feature that was selected was $A18$. This demonstrates that this feature has great predictive power in the data set. The accuracy obtained when using both Gini and entropy DT with the SDTC at a depth level of 1 was 76%. The accuracy is higher than the accuracy achieved by using all 43 features to build a DT model. This fact supports that using feature selection, specifically the SDTC, before building a model can be beneficial.

Comparing the statistics for an entropy DT built with all the features and the SDTC that obtained the best accuracy found in Table 4.3, it is clear the accuracy of the SDTC model is greater than the DT model built with all the features. The sensitivity and specificity of the selected model are also higher. As stated by *Zhu et al.*, in the case of recidivism, a higher sensitivity leads to a model that is better at detecting individuals who might recidivate and a higher specificity leads to a model that is better at detecting who might not recidivate.

A model built with a single feature has limitations when it comes to tool administration. In Table 4.2 the features selected up to a depth level of 3 are shown. This table shows how specifying a larger depth level impacts the number of features selected. It is interesting to look at the common features selected by an SDTC going to a depth level of 3 and an SBC. Gini and entropy DTs selected the same features for the

first 3 levels, so they are combined into one row labeled SDTC. Features that are selected by both methods are A11, A14, A17, A18, A29, A423, A525, and A843. The features selected by both methods could provide more predictive information than the other features in the data set.

In Table 4.3, an SDTC obtained a 76% accuracy, a 33% sensitivity and a 95% specificity compared to an SBC that obtained a 73% accuracy, a 18% sensitivity, and a 98% specificity. An SDTC obtained the best scores at a depth level of 1 on the LSI data set where as an SBC used a set depth level of 3, not allowing for further depth analysis. If one was to compare the scores obtained by an SDTC at a depth level of 3, the accuracy scores of an SDTC and an SBC are within 1% of each other, the SDTC performing slightly worse than the SBC. It is also seen that the SBC did achieve a higher specificity score than the SDTC by 3%, but the SDTC obtained a 15% better sensitivity score than the SBC. For this data set, an SDTC outperforms an SBC because of its ability to select features from depth levels other than 3. By improving the accuracy able to be achieved by the risk assessment tool, not only does it aid in a better prediction, but it also allows the administrator to more accurately allocate intervention methods to prevent the possible recidivism from happening.

4.2.3 LSI-R:SV

The LSI-R:SV data set is described in this section. The description is followed by the results obtained comparing the feature selection methods. Lastly, a discussion of the results is given.

The LSI-R:SV is made up of 8 of the original 43 LSI questions. One of the 8 questions chosen from the original 43 is not from the original screening version, so the LSI-R:SV used in this study is a proxy of the original screening version. A Gini DT is built using only those 8 features and the same 80% training and 20% testing data split that SDTCs used on the full LSI data. By using the same training and testing split, the results of a DT built with the short form features and the selected features can be more validly compared. The selected feature sets used for comparison with the LSI-R:SV set are those obtained from Section 4.2.1.

The results in Table 4.4 compare the validity scores of each a Gini SDTC and an entropy SDTC at depth levels 2 and 3 and the features selected using an SBC with the LSI screening version (LSI-R:SV). Table 4.5 compares the features selected by the various methods at a depth of 3, therefore including the features at depth 2 in the DT. In Table 4.5 each method is given a distinct colour; if a feature is selected by that method, the box is filled in with the designated colour. Row SDTC represents both the Gini and entropy splitting criterion results; the same features were selected by both when using a depth level of 3.

		Depth 1	Depth 2	Depth 3	
	LSI-R:SV	SDTC	SDTC	SBC	SDTC
Accuracy	74.7	76.1	72.4	73.9	73.0
Sensitivity	34.9	33.7	11.2	18.0	13.4
Specificity	92.4	95.0	99.6	98.8	99.6
PPV	67.2	75.1	93.6	87.0	93.9
NPV	76.1	76.2	71.5	73.0	72.0
FPR	7.57	4.98	0.33	1.19	0.38
FNR	65.0	66.2	88.7	81.9	86.5
FDR	32.7	24.8	6.32	12.9	6.06
# Features	8	1	5	15	11

Table 4.4: LSI-R:SV Comparison (%)

	A11	A12	A13	A14	A15	A16	A17	A18	A29	A210	A216	A217	A319	A423	A524	A525	A733	A734	A735	A736	A737	A840	A843
SBC																							
362																							
LSI-R:SV																							

Table 4.5: LSI-R:SV Feature Comparison

The best accuracy obtained out of all the models was 76.128% with the SDTC at a depth level of 1. This depth level only selects a single feature making it hard to compare with the LSI-R:SV. Because of this, the LSI-R:SV is compared to models built using a depth level of 2 or 3. The LSI-R:SV features were used to build a DT with the LSI data set and the model obtained a 74% accuracy as seen in Table 4.4. This was the best accuracy out of the other models built using features from up to depth level 2 and depth level 3. The model that achieved the next highest accuracy of 73% was built using an SBC. The other models were not far behind with accuracies varying from 73% to 72%. The fact that the LSI-R:SV achieved the top accuracy out of all the tested models helps validate that the theoretically chosen set of features that make up the LSI-R:SV are a good set of predictors.

Comparing the sensitivity and specificity of the built models, the models with features selected by the SDTC consistently have a higher specificity than the LSI-R:SV model by at least 7%. They do however, have a lower sensitivity than the LSI-R:SV model. If the administrators were looking for a set of features that more accurately identify the set of individuals who are not likely to re-offend, then the model built using the SDTC would be optimal. If the administrators were looking for a test that more accurately identifies individuals who are likely to re-offend, then the LSI-R:SV set of features may be more optimal.

Looking at the features selected by the various methods in Table 4.5, it is interesting to see the wide variation in features used to build the different models. The Table shows features selected from the SDTCs built using a depth level of 3, therefore including the features that would be selected using a depth level of 2.

The feature selected as the most important by SDTC is $A18$, which is not present in the LSI-R:SV feature set. The features from LSI-R:SV that overlap with the other models are $A12$, $A29$, and $A525$. $A29 and A525$ are selected by all of the models. Feature $A29$ considers if the individual is currently unemployed. Feature $A525$ considers if the individual has friends who are criminals. Using the feature selection information a criminal psychologist may be able to discern further details and improve upon or make alternate forms of the LSI-R:SV.

The LSI-R:SV was able to slightly outperform the other models when considering accuracy. The SDTC models outperformed LSI-R:SV in sensitivity, but not specificity. This analysis brought to light new features that have potential to improve the LSI-R:SV or provide alternate versions of the test.

4.3 Missing Youths Experiments

A classification feature is a feature of a data set on which a prediction model is trained. There is no predesignated classification feature for the missing youths data set. The classification feature chosen to train a model differs depending on what type of prediction is desired from a given model.

The first feature identified as a classification feature from the Missing Youths database is *missing_ again*. This feature has a value of 1 if the individual has gone missing more than once and 0 if not. Using *missing_ again* as a classification feature allows us to train a predictive model to assess the likelihood of an individual to go missing again.

The second feature identified as a classification feature from the Missing Youths database is *gang_ involvement*. This feature has a value of 1 if the individual is believed to have gang involvement and 0 if not. Using *gang_ involvement* as a classification feature allows for the classification model to be trained to assess the likelihood of an individual to be involved in gang activity.

The data were initially received from SPS in the form of multiple linked tables in the Missing Youths Database. The classification features along with a variety of other features associated with the missing youth and their case were pulled from all available tables to make up the missing youth data set used in this thesis. The data were pulled from the SPS Missing Youths Database using mySQL; documentation for mySQL can be found by *Greenspan and Bulger*. The data were combined to contain the features specified in Appendix **??**. Features of the data that do not add to the predictive validity of the model because they are highly individualistic, e.g., surname, g1, g2, and dob, were removed from the data set. All other available data from the Missing Youths Database were used to ensure no bias was introduced by preselecting features by hand. The last step in preparing the data to be used to build the classification model was to encode the string values to numeric values. This does not change what the data represent but rather changes the format in which they are represented and allows for easier manipulation within the algorithms.

The final missing youths data set contained 434 juvenile runaways between September 27, 1971 and February 18, 2016 with 91 distinct features. The feature *missing_ again* contains 171 instances of individuals

who went missing more than once (and accordingly 263 instances of juvenile runaways who only went missing once). The feature *gang_ involvement* contains 71 juvenile runaways with suspected gang involvement (and accordingly 363 instances of no suspected gang involvement).

The goal set to be achieved by building a model on the given data is to be able to provide social workers and officers with a model that can aid in the prediction of children going missing again or missing children who may be involved in gang related activities. It was also important to identify the features that could be more important to answering those two questions. Providing social workers and officers with a set of features that are more important than the others can help to provide a minimum requirement of data features that should be collected to aid in predictions and improving the predictive models.

4.3.1 Results for Classification Feature *missing_ again*

Table 4.6 shows the results for an SDTC and an SBC applied to the missing youths data set with the classification feature *missing_ again*. Table 4.7 are the detailed validity scores for the highest scoring depth level, depth level 1, the scores for a entropy DT built with all the features, and an SBC applied to the data.

Depth	Splitting	# Features Selected	Acc. (%)
1	Gini	2	91.8
	Entropy	2	91.8
2	Gini	7	88.3
	Entropy	8	86.0
3	Gini	13	87.2
	Entropy	15	89.5
4	Gini	17	86.0
	Entropy	16	83.7
All Features	Gini	90	82.5
	Entropy	90	84.8
SBC	Entropy	13	84.8

Table 4.6: SDTC Results for *missing_ again*

	Entropy & Gini	All Features Entropy	SBC
Accuracy	91.8	84.8	84.8
Sensitivity	79.4	73.5	70.5
Specificity	100.	92.3	94.2
PPV	100.	86.2	88.8
NPV	88.1	84.2	83.0
FPR	0.00	7.69	5.76
FNR	20.5	26.4	29.4
FDR	0.00	13.7	11.1

Table 4.7: Statistics for the Best Selected Model and the Original Full Model in Percent (%)

Below are the features selected at depth level 1 through 3 for Gini and entropy SDTCs. At each level, the new features that were added are bolded, leaving the features that were selected in previous levels normal.

The features selected at depth 1 by both a Gini and an entropy SDTC are:

- **missing_period**
- **missing_per_year**

At depth level 2, a Gini SDTC selected the features:

- missing_period
- missing_per_year
- **age_last**
- **y_coordinate**
- **disability**
- **gang_name**
- **height**
- **eye_color**

At depth level 3, a Gini SDTC selected the features:

- missing_period
- missing_per_year
- age_last
- y_coordinate
- disability
- gang_name
- height
- **family_violence**
- **location**
- **object_of_theft3**
- **facial_hair_color**
- **drivers_licence**
- **weight**

At depth level 2, an entropy SDTC selected the features:

- missing_period
- missing_per_year
- **age_group**
- **age_last**
- **date_located**
- **y_coordinate**
- **disability**
- **gang_name**

At depth level 3, an entropy SDTC selected the features:

- missing_period
- missing_per_year
- age_last
- y_coordinate
- disability
- gang_name
- **coat_color1**
- **location**
- **occ_date**
- **vehicle_target**
- **citizenship**
- **drivers_licence**
- **hair_color**
- **height**
- **weight**

4.3.2 Discussion for Classification Feature *missing_again*

Looking at the results produced for the classification feature *missing_again* in Table 4.6, the SDTC was able to obtain a higher accuracy than a DT while reducing the number of features needed to build the model. The SDTC model was able to obtain an 91% accuracy and increased both sensitivity and specificity when compared to the DT model. The best accuracy was achieved with only two features for both a Gini and entropy based SDTC. Those features were *missing_period* and *missing_per_year*. The features selected were verified by a psychologist at SPS. It was stated that the variables selected seemed plausible and matched the experiences the psychologist has had working with missing persons, and specifically missing youths, cases. This affirmation helps validate the feature selection process an SDTC uses by demonstrating that an SDTC produces results that match the expectations of an expert in the field.

In the best case, an SDTC was able to successfully predict whether an individual would go missing again with over 91% accuracy, as seen in Table 4.6. It also identified the most influential features in such situations. This information may be useful to officers to consider when deciding if a professional should intervene in cases where an individual is at a high risk to go missing again.

SDTC obtained a 91% accuracy, a 79% sensitivity, and a 100% specificity compared to a SBC that obtained a 84% accuracy, a 70% sensitivity, and a 94% specificity. The depth level that SDTCs performed that best at was 1. Comparing the results obtained by an SDTC at a depth level of 3 with the SBC results, an SDTC obtains a higher accuracy using both a Gini and entropy DTs. Using a Gini DT the SDTC obtain a 14% higher accuracy and using an entropy DT the SDTC obtain a 16% higher accuracy.

4.3.3 Results for Classification Feature *gang_ involvement*

Table 4.8 shows the results for an SDTC and an SBC applied to the missing youths data set with the classification feature *gang_ involvment*. Table 4.9 shows all of the validity scores for the highest scoring depth level and splitting criterion, Gini at depth level 2, the scores for a Gini DT built with all features, and an SBC applied to the data.

Depth	Splitting	# Features Selected	Acc. (%)
1	Gini	2	84.8
	Entropy	3	87.2
2	Gini	11	88.3
	Entropy	8	84.8
3	Gini	18	88.3
	Entropy	13	83.7
4	Gini	25	81.3
	Entropy	19	81.3
All Features	Gini	90	83.7
	Entropy	90	82.5
SBC	Entropy	15	89.5

Table 4.8: SDTC Results for *gang_ involvment*

	Depth 2 Gini	All Features Gini	SBC
Accuracy	88.3	83.7	89.5
Sensitivity	50.0	50.0	71.4
Specificity	95.8	90.2	93.0
PPV	70.0	50.0	66.6
NPV	90.7	90.2	94.3
FPR	4.16	9.72	6.94
FNR	50.0	50.0	28.5
FDR	30.0	50.0	33.3

Table 4.9: Statistics for the Best Selected Model and the Original Full Model in Percent (%)

Below are the features selected at depth level 1 through 3 for Gini and entropy SDTCs. At each level, the new features that were added are bolded, leaving the features that were selected in previous levels normal.

At depth level 1, a Gini SDTC selected the features:

- **employer**
- **height**

At depth level 2, a Gini SDTC selected the features:

- employer
- height
- **occupation**
- **sex**
- **cause_of_absence**
- **missing_period**
- **missing_per_year**
- **place_last_seen**
- **gang_type**
- **place_name**
- **week_day**

At depth level 3, a Gini SDTC selected the features:

- employer
- height
- occupation
- sex
- cause_of_absence
- missing_period
- missing_per_year
- place_last_seen
- gang_type
- place_name
- week_day
- **age_first**
- **cyber_crime_type**
- **object_of_theft1**
- **location**
- **occ_num**
- **vehicle_type**
- **hair_style**

At depth level 1, an entropy SDTC selected the features:

- **country_of_birth**
- **employer**
- **height**

At depth level 2, an entropy SDTC selected the features:

- country_of_birth
- employer
- height
- **occupation**
- **location**
- **week_day**
- **x_coordinate**
- **y_coordinate**

At depth level 3, an entropy SDTC selected the features:

- country_of_birth
- employer
- height
- occupation
- location
- week_day
- x_coordinate
- y_coordinate
- **age_first**
- **age_last**
- **place_name**
- **vehicle_type**
- **caution**

4.3.4 Discussion for Classification Feature *gang_involvement*

Looking at the results produced for the classification feature *gang_involvement* in Table 4.8, the SDTC was able to obtain a higher accuracy than a DT while reducing the number of features needed to build the model. The best accuracy was achieved using the Gini splitting criterion with only eleven features. Those features were *employer*, *height*, *occupation*, *sex*, *cause_of_absence*, *missing_period*, *missing_per_year*, *place_last_seen*, *gang_type*, *place_name*, and *week_day*.

An SDTC was able to obtain an 88% accuracy and increased specificity when compared to the DT model as seen in Table 4.8. Looking at the results obtained for the SDTC, the false negative rate was high, being no better than random selection. An SDTC is able to identify people who are likely to be involved with gangs with 50% sensitivity, leading to this 50% false negative rate. The false negative rate measures how often people who are identified as *not* likely to be involved in gangs are actually involved in gangs. This high false negative rate could be because of the small sample size of individuals who did have gang involvement. The sample size of individuals with gang involvement is 71 out of the total 434 cases. When 80% of those samples are used in the training data set, relatively few samples remain for testing. This could lead to overly low false positive rates and overly high false negative rates.

In the best case, SDTC was able to successfully predict whether an individual would go missing again with over 88% accuracy. It also identified the most influential features in such situations. These models can be used by officers to determine if a missing youth is likely to have gang affiliations, knowing this could change how the case is approached and the safety measures the officers may take.

An SDTC obtained an 88% accuracy, a 50% sensitivity, and a 95% specificity compared to an SBC that obtained an 89% accuracy, a 71% sensitivity, and a 93% specificity. The SDTC performed the best using a Gini DT at a depth level of 2. Comparing the SDTC at a depth level of 3 to the SBC, the SBC did outperform both a Gini and an entropy based SDTC. An SBC outperformed a Gini SDTC by 1% and outperformed an entropy SDTC by 6%.

4.4 CAMCI Experiments

The original CAMCI data set was obtained from Psychological Software Tools, Inc. (Pittsburgh, PA). The data set used in this thesis is a normalized version of this data set received from the Psychology Department at the University of Saskatchewan. The participants in the study were all over the age of 18 and had at least an eighth-grade education. If there was a previous diagnosis of a cognitive disorder of impairment including, alcohol abuse, stroke, or serious head injury, the participant was excluded from the data set. The participant also had to be able to read and write. The initial data set contained 1,173 participants, but many were eliminated because of the above reasons or missing points in their data entry. The final number of participants included in the data set used was 580.

The data are comprised of 33 features and a classifier. The classifier has a value of 1 or 0, 1 if the participant is diagnosed by a professional to have mild cognitive impairment or 0 otherwise. The 33 features included sex, gender, age, years of education, 13 reaction time values, and 16 test scores. The full description of each cognitive test feature is found in Appendix B. The data used have been normalized to negate the continuous aspect of the reaction time features. The normative data made it easier to apply machine learning techniques to.

The goal of building a model on the given data is to be able to provide medical practitioners with a classification model that does not miss an individual who may have early stages of Alzheimer's disease or dementia. This means that it is more important for the model provided to be highly sensitive, not classifying individuals who do have early stages of Alzheimer's disease and dementia as not having it. It was also important to identify the features that could be more important to aiding in the prediction of early stages of Alzheimer's disease and dementia. By doing this, a shortened test could be provided to practitioners to reduce the time resources required to administer the CAMCI tool.

4.4.1 Results

Table 4.10 shows the results of an SDTC, a DT, and an SBC applied to the CAMCI data. The table details the depth, the type of tree, the amount of time it would take to complete the test with the specific features selected, and the accuracy. In the "Seconds to Complete" column there are two values, the first is the amount of time to complete the test in seconds and the second is the number of features selected in brackets. The time it takes to complete the test is calculated by adding together the time on average it takes to complete each test component. A test component is added to the test if one of the features associate with it is selected. If two features from a single component are selected the time added would be the same if only one feature associated with the component was selected. Remembering that the random seed value used to shuffle the data can impact how the data are split, a standardized value of 362 was used. Having a standardized random seed value ensures that the splits in the training data are the same for each the Gini and entropy SDTCs.

Depth	Splitting	Seconds to Complete (s)	Acc. (%)
1	Gini	848 (5)	57.6
	Entropy	848 (5)	57.6
2	Gini	1149 (11)	58.6
	Entropy	1300 (12)	59.6
3	Gini	1333 (18)	58.6
	Entropy	1356 (16)	59.6
4	Gini	1484 (20)	59.6
	Entropy	1356 (20)	51.9
All Features	Gini	1484	53.8
	Entropy	1484	56.7
SBC	Entropy	1198 (13)	61.5

Table 4.10: SDTC Results for CAMCI

Table 4.11 shows all of the validity scores for the highest scoring depth level, a depth of 2, and the entropy splitting criterion compared with the scores for an entropy DT trained using all the features in the CAMCI data set and an SBC.

	Depth 2 Entorpy	All Features Entropy	SBC
Accuracy	59.6	56.7	61.5
Sensitivity	75.8	70.6	58.6
Specificity	39.1	39.1	65.2
PPV	59.4	59.4	68.0
NPV	51.4	51.4	55.5
FPR	60.8	60.8	34.7
FNR	24.1	29.3	41.3
FDR	38.8	40.5	32.0

Table 4.11: Statistics for the Best Selected Model and the Original Full Model in Percent (%)

Below are the features selected at depth level 1 through 3 for the Gini and entropy SDTCs. At each level, the new features that were added are bolded, leaving the features that were selected in previous levels normal.

At depth level 1, a Gini SDTC selected the features:

- **RPNmCorRspAC**
- **RPNmIncRspAC**
- **WRNmCorRspAC**
- **DR1stTriRT**
- **VEITNmCorAC**

At depth level 2, a Gini SDTC selected the features:

- **Age**
- **YrsEdu**
- **SRAllRspDRT**
- **RPDPrimeAC**
- RPNmIncRspAC
- WRNmCorRspAC
- DR1stTriRT
- **DRMaxSpanAC**
- **VEATMAC**
- **VEINTAC**
- VEITNmCorAC

At depth level 3, a Gini SDTC selected the features:

- Age
- YrsEdu
- SRAllRspDRT
- **SRCorRspDRT**
- **RPCorRspDRT**
- RPDPrimeAC
- RPNmCorRspAC
- RPNmIncRspAC
- WRNmCorRspAC
- **GN1NmCorAC**
- **GN2CorRspART**
- **DF1stTraRT**
- DR1stTriRT
- DRMaxSpanAC
- VEATMAC
- **VEINNmCorAC**
- VEINTAC
- VEITNmCorAC

At depth level 1, an entropy SDTC selected the features:

- **RPDPrimeAC**
- **RPNmCorRspAC**
- **WRNmCorRspAC**
- **DR1stTreRT**
- **VEITNmCorAC**

At depth level 2, an entropy SDTC selected the features:

- RPDPrimeAC
- RPNmIncRspAC
- WRNmCorRspAC
- DR1stTreRT
- VEITNmCorAC
- **Age**
- **YrsEdu**
- **DF1stTriRT**
- **DRMaxSpanAC**
- **VRCorRspDRT**
- **VEATMAC**
- **VEINTAC**

At depth level 3, an entropy SDTC selected the features:

- RPDPrimeAC
- **RPCorRspDRT**
- RPNmIncRspART
- WRNmCorRspAC
- DR1stTreRT
- VEITNmCorAC
- Age
- YrsEdu
- DF1stTriRT
- DRMaxSpanAC
- VEATMAC
- VEINTAC
- **SRCorRspDRT**
- **RPNmCorRspAC**
- **RPNmIncRspAC**
- **VEINNmCorAC**

4.4.2 Discussion

Looking at the results produced by the CAMCI data set being used on both a DT and SDTC, an SDTC achieved a higher accuracy. The best accuracy, 59%, was obtained by an entropy SDTC model using a depth level of 2. Other variations of the SDTC model achieved similar scores, but the entropy SDTC going to a depth level of 2 had the smallest feature set of all the similar accuracy scores. This top model was able to select a feature set that would only take 21.67 minutes to complete, compared to the original set of features that took 24.73 minutes that obtained an accuracy of 56% using a DT model. The shortest test completion, and therefore test with the fewest features, obtained a completion time of 14.13 minutes. This short test has an accuracy of 58%, still achieving an increase in accuracy when compared to the DT model built from all of the features.

Analyzing the results shows that an SDTC is superior to a DT for this data set. From this analysis, two new shortened test could be provided to practitioners and patients, one that maximizes accuracy and reduces the time to complete the test by a small amount, or one that reduces the time to take the test by a large amount, but increases the accuracy by only a small amount. Using either new set of test questions enhances the ability to diagnosis mild cognitive impairment using the CAMCI test.

On the CAMCI data set an SDTC obtained a 59% accuracy, a 75% sensitivity, and a 39% specificity compared to an SBC that obtained a 61% accuracy, a 58% sensitivity, and a 65% specificity. The SBC outperformed the SDTC on accuracy and specificity, but achieved a significantly lower sensitivity score than the SDTC. Depending on what a clinician is hoping to derive from the model, both the SDTC and the SBC models have usefulness.

4.5 SBC vs. SDTC Discussion

The results in Table 4.12 are the validity scores for the SBC applied to each of the data set.

Selective Bayesian Results	Accuracy	Sensitivity	Specificity	PPV	NPV	FPR	FNR	FDR
LSI	73.9	18.0	98.8	87.0	73.0	1.19	81.9	12.9
Missing Youths Missing Again	84.8	70.5	94.2	88.8	83.0	5.76	29.4	11.1
Missing Youths Gang Involvement	89.5	71.4	93.0	66.6	94.3	6.94	28.5	33.3
CAMCI	61.5	58.6	65.2	68.0	55.5	34.7	41.3	32.0

Table 4.12: SBC Results in Percent (%)

The SDTC method was designed based off of the simpler SBC method. Comparing the results obtained by both on the same sets of data provides useful insight into possible strengths and weaknesses of each method. The initial observation is that on three of the four data sets used an SDTC outperforms or performs equally to an SBC in terms of accuracy.

Knowing that the SBC selects features using only a depth level of 3, it makes sense to compare the performance of itself and an SDTC at this depth level. It is seen that the SDTC does not always obtain a higher accuracy than the SBC when using a depth level of 3. Only using the missing youths data set with a classification feature of *missing_ again* does the SDTC out perform an SBC at a depth level of 3. When allowing an SDTC to select other depth levels, it can achieve better overall validity scores than an SBC on 3 out of the 4 data sets tested. This finding supports the addition of flexibility in selecting various depth levels that the SDTC provides. Being able to choose the depth level features are selected up to allows for improved validity scores and ultimately a better model.

The flexibility of being able to choose the splitting criterion used in the DT methods also adds another element of fine tuning to the SDTC. In some cases, Gini and entropy DTs do not perform equally on the data sets. For example, on the missing youths data set when using the *gang_ involvement* classification feature, the Gini splitting criterion substantially increased the accuracy obtained by the model when compared to the entropy splitting criterion results. The SBC only uses the entropy splitting criterion, potentially missing the opportunity to obtain a higher accuracy with an alternate splitting criterion. Different splitting criteria may also select different features. This is seen in a few instances, one being where a Gini SDTC at a depth level of 1 selects *employer* and *height* from the *gang_ involvement* missing youths data set, whereas an entropy SDTC at a depth level of 1 selects *employer*, *height*, and *country_ of_ birth*. An SDTC can aid in analyzing various subsets of features obtained when using different splitting criteria of DTs for feature selection.

5 Conclusion and Suggestions for Future Work

5.1 Conclusion

This thesis proposed a new features selection technique, the SDTC. This new technique is derived from the existing SBC and uses machine learning models to select the features and provide a predictive model. An SDTC uses DTs in two ways, to select features and to provide the final predictive model. The goal was to investigate whether SDTCs are a valid form of feature selection that offer beneficial improvements and variations to an SBC. By achieving this goal, not only does it further the field of machine learning, but also each field of application among the three data sets used.

Machine learning is a powerful tool that, when used properly, can be beneficial to a variety of problems (*Pang et al.*, 2002; *Nguyen and Armitage*, 2008; *Carbonell et al.*, 1983). Feature selection can be used in conjunction with machine learning to improve the models built. A way feature selection improves model building is by helping to clean the data; various studies demonstrate why data cleaning is important and how feature selection is used to aid in this task (*Batista and Monard*, 2003; *Yang et al.*, 2003; *Salas-Gonzalez et al.*, 2010; *Mugunthadevi et al.*, 2011). The main machine learning method used in this thesis was DTs. It is important to understand how DTs are built to fully understand how the SDTC selects features and how the classification model is built. The SDTC models were built using three different data sets. The first was a data set comprised of LSI scores and features. The second was a data set comprised of missing youths data. The second data set was used twice with two different classification features, *missing_again* and *gang_involvement*. The third data set was comprised of CAMCI features and a diagnosis on whether or not the patient had mild cognitive impairment made by a medical practitioner. The results for the models built are given and used to validate a series of objectives presented in this thesis.

Contributions to the machine learning and feature selection fields align with the three main objectives of this thesis. The first was to implement the SDTC and test its validity using various data sets. The SDTC was implemented using the Python programming language and tested on three different data sets. Comparing the results obtained to a regular DT, SDTCs outperformed DTs on all of the data sets. This finding supports that data cleaning and feature selection are an important step in machine learning. It also demonstrates that using feature selection and more specifically SDTCs helps to improve the quality of accuracy, sensitivity, and specificity of classification models. The results also show that using different splitting criteria for the DTs is advantageous. When using the missing youths data set and the classification feature *gang_involvement* an SDTC using a Gini splitting criterion outperformed an SDTC using an entropy splitting criterion. Without

the ability to change the splitting criteria used, there is a risk of obtaining a classification model with a decreased accuracy.

The second objective was to compare an SDTC to an SBC. This comparison demonstrates how the SDTC improves upon the SBC method, yet another beneficial contribution to machine learning and feature selection fields. Three different data sets were used to build an SDTC and an SBC and the results of each were compared. In 2 out of the 4 data sets used, the SDTC outperforms an SBC. On the missing youths data set using the *gang_involvement* classification feature, an SDTC and an SBC perform comparably. On the CAMCI data set, an SBC did outperform an SDTC. The SDTC is more customizable than the SBC, allowing it to select various sizes of feature sets and use different splitting criteria for the DTs. In all 3 of the cases where SDTCs outperformed or performed comparably to SBCs, it was not only because of the added flexibility, but also because of the use of a different classification model. This is seen in the missing youths data set when using the classification feature *gang_involvement*. The SBC performed comparably to an SDTC using a Gini splitting criterion, but it performed better than an SDTC using an entropy splitting criterion. When comparing an SDTC and an SBC built using a depth level of 3, an SDTC did not outperform an SBC, but the SDTC did outperform an SBC when using other depth levels of features. This clearly shows that the added flexibility of depth level selection gives the SDTC an advantage over the SBC.

A methodological improvement the SDTC has compared to the SBC is the way it splits up the training data for the feature selection step. An SBC splits the training data by taking five 10% random samples with replacement to build the DTs for feature selection. This approach does not utilize the entire data set and has the chance of using data points in more than one of the five DTs. This overlap in data used could result in an over emphasis on a single datum, influencing the results of the feature selection process. The SDTC improves this selection process by utilizing the entire training data set and removing the possibility of data overlap. An SDTC shuffles and splits the training data into five pieces and uses those five distinct subgroups of the training data to build the DTs used for the feature selection step.

The third objective was to demonstrate an application of the SDTC and compare the features selected to a pre-existing set of selected features. Comparing the features selected by SDTC to an established method helps validate the importance of developing new feature selection and classification techniques. This was done by applying the SDTC to the LSI data and comparing the features the SDTC selected with the features of the LSI-R:SV. The LSI-R:SV is comprised of 8 of the LSI features. LSI-R:SV has not been empirically validated, but was created theoretically by psychologists and correctional professionals. The results showed that there was a 50% overlap of the features in the LSI-R:SV with features selected from the various runs of the SDTC. Using the features in the LSI-R:SV to build a DT it obtained a slightly higher accuracy than the SDTC was able to obtain. The SDTC was able to obtain a higher specificity score than the LSI-R:SV DT. This analysis did help validate the LSI-R:SV by demonstrating that there is an overlap in features and showed that the LSI-R:SV does have a slight superiority in accuracy over an SDTC. This analysis also provided alternative shortened LSI tests that do better than the LSI-R:SV at more accurately identifying the set of individuals

who are not likely to re-offend.

The main contribution of this thesis was the development of the SDTC method. By applying the SDTC to the various data sets, shortened forms of the pre-existing tests were derived. These short form tests could be used by professionals, reducing administration time and improving the accuracy of the overall test. In the case of the missing youths data and the CAMCI data, there were no short form tests and this contribution could be valuable to the administrators. By validating that the SDTC feature selection and classification method improves upon a standard DT method and the SBC, it shows that SDTC is a positive contribution to the fields of machine learning and feature selection.

5.2 Future Work

An aspect of the SDTC that was not explored to the fullest was the number of DTs used during the feature selection step. Currently, both an SDTC and an SBC split the training data into five pieces and build five DTs to select the features from. This number "five" has not been validated. *Ratanamahatana and Gunopulos*, the authors who presented the SBC, stated that they used five DTs because it seemed to work well for the data sets used in their experiments. They did not report other results using different numbers of DTs or explain whether they tested using other numbers of DTs for the feature selection step. To further improve this algorithm, further testing on how many DTs for the feature selection process are optimal could be done. From the models built using the SDTC in this thesis, the more data used within the five DTs built, the more similar the DTs looked in terms of structure and where the features were placed. Two questions to consider for the feature selection step in SDTCs would be: Is this similarity what should be striven for when splitting data for the feature selection process; does splitting up the data into more pieces, causing the DTs to be built with less data and possibly greater variation in feature placement, better for the feature selection process?

An intriguing aspect of this thesis work was the idea of overlapping selected features discussed in Section 4.2.1. By comparing various combinations of depth levels, splitting criteria, and random states in the SDTC there were features that were selected by all combinations. This subset of the selected features could prove to have even greater classification potential than a single feature selection method has on its own. Continuing down this line of thinking, it would be interesting to compare other features selected by various feature selection methods. With information gained from various methods, one could create a weighted set of selected features. This could be done by assigning a lower weight to features selected by only one feature selection method, and increase the weight the selected feature has in the set when it is selected by more feature selection methods. In future, building classification models with weighted sets of overlapping features from various feature selection methods could lead to a more intelligent feature selection method and a better classification model.

Similarly, it would be interesting to combine various sets of features selected in Section 4.2.3 that make up a shortened screening version of the LSI test. The LSI:R-SV obtained the best accuracy out of the methods

tested, but adding or substituting in some of the other top features not in the LSI:R-SV might affect the validity scores obtained in a positive way.

The data sets used to build the models were all centered around social sciences. These types of data sets can be very different than other data sets not only in terms of content but also in terms of characteristics of the data set. For example, a data set on genetics can have a large number of dimensions or features and is scored by a computer. The data being working with in this thesis does not have over 100 dimensions and is entered or recorded by a human. Both of these aspects cause the two types of data sets to be very different. Testing the SDTC on other types of data in the future would be critical in determining if it can be applied to a variety of problems.

Currently, the SDTC provides a boolean classification, yes or no. This is a useful classification, but maybe not as useful as it could be. In future, it would be beneficial to add in a percentage value to the classification given. The percentage would describe how homogeneous the classification was. A classification is calculated from the classification feature values of the subset of data contained in the leaf or classification node. Sometimes the entire subset does not have one uniform classification value and can therefore be given as a percentage rather than a boolean yes or no value.

Another idea stumbled upon through this thesis work was the idea of ensemble methods (*Dietterich*, 2000). Ensemble methods utilize the strengths of other classification methods to enhance the predictive capabilities of a model. The aspect of ensemble methods that was particularly intriguing was the concept of adding the results of multiple classification methods built from a data set to the data set itself as additional features. It would be intriguing to add this step before and after the feature selection method to analyze the impact ensemble methods have on feature selection and on the final classification method. By studying this, one could add onto the steps already performed by SDTCs, enhancing the method further.

Bibliography

Albert, M. S., S. T. DeKosky, D. Dickson, B. Dubois, H. H. Feldman, N. C. Fox, A. Gamst, D. M. Holtzman, W. J. Jagust, R. C. Petersen, et al. (2011), The diagnosis of mild cognitive impairment due to Alzheimer's disease: Recommendations from the national institute on Aging-Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease, *Alzheimer's & dementia*, *7*(3), 270–279.

Andrews, D. A., and J. Bonta (1998), The Level of Service Inventory–Revised: Screening Version ., *Tech. rep.*, Toronto: Multi-Health Systems.

Andrews, D. A., and J. Bonta (2010), *The psychology of criminal conduct*, Routledge.

Andrews, D. A., J. Bonta, and J. S. Wormith (1995), Level of Service Inventory–Ontario revision (LSI-OR): Interview and scoring guide, *Toronto, Canada: Ontario Ministry of the Solicitor General and Correctional Services*.

Andrews, D. A., J. Bonta, and J. S. Wormith (2006), The recent past and near future of risk and/or need assessment, *Crime & Delinquency*, *52*(1), 7–27.

Batista, G. E., and M. C. Monard (2003), An analysis of four missing data treatment methods for supervised learning, *Applied artificial intelligence*, *17*(5-6), 519–533.

Bonny, E., L. Almond, and P. Woolnough (2016), Adult missing persons: Can an investigative framework be generated using behavioural themes?, *Journal of Investigative Psychology and Offender Profiling*, *13*(3), 296–312.

Breiman, L. (2001), Random forests, *Machine learning*, *45*(1), 5–32.

Brookmeyer, R., S. Gray, and C. Kawas (1998), Projections of Alzheimer's disease in the United States and the public health impact of delaying disease onset., *American journal of public health*, *88*(9), 1337–1342.

Brookmeyer, R., E. Johnson, K. Ziegler-Graham, and H. M. Arrighi (2007), Forecasting the global burden of Alzheimer's disease, *Alzheimer's & dementia*, *3*(3), 186–191.

Brownlee, J. (2016a), An introduction to feature selection machine learning mastery, http://machinelearningmastery.com/an-introduction-to-feature-selection/, accessed: 2017-05-24.

Brownlee, J. (2016b), Feature selection to improve accuracy and decrease training time machine learning mastery, http://machinelearningmastery.com/feature-selection-to-improve-accuracy-and-decrease-training-time/, accessed: 2017-05-24.

Buntine, W., and T. Niblett (1992), A further comparison of splitting rules for decision-tree induction, *Machine Learning*, *8*(1), 75–85, doi:10.1007/BF00994006.

Cai, L., and Y. Zhu (2015), The challenges of data quality and data quality assessment in the big data era, Ubiquity Press.

Carbonell, J. G., R. S. Michalski, and T. M. Mitchell (1983), An overview of machine learning, in *Machine Learning, Volume I*, pp. 3–23, Elsevier.

Cawley, G. C., and N. L. Talbot (2010), On over-fitting in model selection and subsequent selection bias in performance evaluation, *Journal of Machine Learning Research*, *11*(Jul), 2079–2107.

Chandrashekar, G., and F. Sahin (2014), A survey on feature selection methods, *Computers & Electrical Engineering*, *40*(1), 16–28.

Correctional Service of Canada, Policy and Research Sector, Research (2015), Forum on corrections research, http://www.csc-scc.gc.ca/research/forum/special/espe_a-eng.shtml.

Dana, A.-D., and A. Alashqur (2014), Using decision tree classification to assist in the prediction of Alzheimer's disease, in *Computer Science and Information Technology (CSIT), 2014 6th International Conference on*, pp. 122–126, IEEE.

D'andrade, A., M. J. Austin, and A. Benton (2008), Risk and safety assessment in child welfare: Instrument comparisons, *Journal of Evidence-Based Social Work*, *5*(1-2), 31–56.

Dietterich, T. G. (2000), Ensemble methods in machine learning, in *International workshop on multiple classifier systems*, pp. 1–15, Springer.

Domingos, P. (2012), A few useful things to know about machine learning, *Communications of the ACM*, *55*(10), 78–87.

Elkan, C. (1997), Boosting and Naive Bayesian learning, *Tech. rep.*, Technical Report CS97-557, University of California, San Diego.

Erman, J., A. Mahanti, and M. Arlitt (2006), Internet traffic identification using machine learning, in *Global Telecommunications Conference, 2006. GLOBECOM'06. IEEE*, pp. 1–6, IEEE.

Eshelman, L. J. (1991), The chc adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination, in *Foundations of genetic algorithms*, vol. 1, pp. 265–283, Elsevier.

Evans, D. A., H. H. Funkenstein, M. S. Albert, P. A. Scherr, N. R. Cook, M. J. Chown, L. E. Hebert, C. H. Hennekens, and J. O. Taylor (1989), Prevalence of Alzheimer's disease in a community population of older persons: higher than previously reported, *Jama*, *262*(18), 2551–2556.

Fisher, R. A. (1936), The use of multiple measurements in taxonomic problems, *Annals of eugenics*, *7*(2), 179–188.

Freund, Y., and R. E. Schapire (1997), A decision-theoretic generalization of on-line learning and an application to boosting, *Journal of computer and system sciences*, *55*(1), 119–139.

Fyfe, N. R., O. Stevenson, and P. Woolnough (2015), Missing persons: the processes and challenges of police investigation, *Policing and Society*, *25*(4), 409–425.

Girard, L., and J. S. Wormith (2004), The predictive validity of the Level of Service Inventory-Ontario Revision on general and violent recidivism among various offender groups, *Criminal Justice and Behavior*, *31*(2), 150–181.

Gray, K. R., P. Aljabar, R. A. Heckemann, A. Hammers, D. Rueckert, A. D. N. Initiative, et al. (2013), Random forest-based similarity measures for multi-modal classification of Alzheimer's disease, *NeuroImage*, *65*, 167–175.

Greenspan, J., and B. Bulger (2001), *MySQL/PHP database applications*, John Wiley & Sons, Inc.

Gunn, S. R., et al. (1998), Support vector machines for classification and regression, *ISIS technical report*, *14*(1), 5–16.

Kohavi, R. (1996), Scaling up the accuracy of Naive-Bayes classifiers: a decision-tree hybrid., in *KDD*, vol. 96, pp. 202–207, Citeseer.

Kotsiantis, S. B., I. Zaharakis, and P. Pintelas (2007), Supervised machine learning: A review of classification techniques, *Emerging artificial intelligence applications in computer engineering*, *160*, 3–24.

Kubat, M. (2015), in *An introduction to machine learning*, vol. 681, pp. 19–41, Springer.

Li, T., C. Zhang, and M. Ogihara (2004), A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression, *Bioinformatics*, *20*(15), 2429–2437.

Liaw, A., M. Wiener, et al. (2002), Classification and regression by randomforest, *R news*, *2*(3), 18–22.

Lumley, T., et al. (2004), Analysis of complex survey samples, *Journal of Statistical Software*, *9*(1), 1–19.

Mallick, P. K. (2015), *Research Advances in the Integration of Big Data and Smart Computing*, IGI Global.

Marsland, S. (2015), *Machine learning: an algorithmic perspective*, CRC press.

McCullagh, P., and J. A. Nelder (1989), *Generalized linear models*, vol. 37, CRC press.

McKhann, G. M., D. S. Knopman, H. Chertkow, B. T. Hyman, C. R. Jack Jr, C. H. Kawas, W. E. Klunk, W. J. Koroshetz, J. J. Manly, R. Mayeux, et al. (2011), The diagnosis of dementia due to Alzheimer's disease: Recommendations from the National Institute on Aging-Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease, *Alzheimer's & dementia*, *7*(3), 263–269.

Meisner, E. (2003), Naive Bayes Classifier example, http://www.inf.u-szeged.hu/~ormandi/ai2/06-naiveBayes-example.pdf.

Melnychuk, M. C., E. Peterson, M. Elliott, and R. Hilborn (2017), Fisheries management impacts on target species status, *Proceedings of the National Academy of Sciences*, *114*(1), 178–183.

Michalski, R. S., J. G. Carbonell, and T. M. Mitchell (2013), *Machine learning: An artificial intelligence approach*, Springer Science & Business Media.

Mugunthadevi, K., S. Punitha, M. Punithavalli, and K. Mugunthadevi (2011), Survey on feature selection in document clustering, *International Journal on Computer Science and Engineering*, *3*(3), 1240–1241.

Nasreddine, Z. S., N. A. Phillips, V. Bédirian, S. Charbonneau, V. Whitehead, I. Collin, J. L. Cummings, and H. Chertkow (2005), The Montreal Cognitive Assessment, MoCA: a brief screening tool for mild cognitive impairment, *Journal of the American Geriatrics Society*, *53*(4), 695–699.

Nguyen, T. T., and G. Armitage (2008), A survey of techniques for internet traffic classification using machine learning, *IEEE Communications Surveys & Tutorials*, *10*(4), 56–76.

NPIA (2011), Missing persons: data and analysis 2009/10, NPIA Bramshill.

Oraji, R. (2016), A machine learning generalization of LSI-OR, Master's thesis, University of Saskatchewan.

Ozkan, T. (2017), Predicting recidivism through machine learning, ProQuest Dissertations Publishing.

Pang, B., L. Lee, and S. Vaithyanathan (2002), Thumbs up?: sentiment classification using machine learning techniques, in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pp. 79–86, Association for Computational Linguistics.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. (2011), Scikit-learn: Machine learning in Python, *Journal of machine learning research*, *12*(Oct), 2825–2830.

Quinlan, J. R. (1986), Induction of decision trees, *Machine learning*, *1*(1), 81–106.

Quinlan, J. R. (1993), *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

R Core Team (2014), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.

Raileanu, L. E., and K. Stoffel (2004), Theoretical comparison between the Gini index and information gain criteria, *Annals of Mathematics and Artificial Intelligence*, *41*(1), 77–93.

Ratanamahatana, C. A., and D. Gunopulos (2002), Scaling up the Naive Bayesian classifier: Using decision trees for feature selection, Citeseer.

Russell, J. (2015), Predictive analytics and child protection: Constraints and opportunities, *Child Abuse & Neglect*, *46*(C), 182–189.

Salas-Gonzalez, D., J. M. Górriz, J. Ramírez, M. López, I. Alvarez, F. Segovia, R. Chaves, and C. Puntonet (2010), Computer-aided diagnosis of Alzheimer's disease using support vector machines and classification trees, *Physics in Medicine & Biology*, *55*(10), 2807.

Saxton, J., L. Morrow, A. Eschman, G. Archer, J. Luther, and A. Zuccolotto (2009), Computer assessment of mild cognitive impairment, *Postgraduate medicine*, *121*(2), 177–185.

Sayad, S. (2010–2017), An introduction to data mining: Naive Bayesian, *http* : *//www.saedsayad.com/naive_bayesian.htm*, accessed: 2017-06-12.

Sebastiani, F. (2002), Machine learning in automated text categorization, *ACM computing surveys (CSUR)*, *34*(1), 1–47.

Sledjeski, E. M., L. C. Dierker, R. Brigham, and E. Breslin (2008), The use of risk assessment to predict recurrent maltreatment: A classification and regression tree analysis (CART), *Prevention science*, *9*(1), 28–37.

Sperling, R. A., P. S. Aisen, L. A. Beckett, D. A. Bennett, S. Craft, A. M. Fagan, T. Iwatsubo, C. R. Jack Jr, J. Kaye, T. J. Montine, et al. (2011), Toward defining the preclinical stages of Alzheimer's disease: Recommendations from the National Institute on Aging-Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease, *Alzheimer's & dementia*, *7*(3), 280–292.

SPSS, IBM (2013), SPSS statistical software, *Armonk, NY: IBM Corporation*.

Su, Y., T. Murali, V. Pavlovic, M. Schaffer, and S. Kasif (2003), RankGene: identification of diagnostic genes based on expression data, *Bioinformatics*, *19*(12), 1578–1579.

Thirion, B., E. Duschenay, V. Michel, G. Varoquaux, O. Grisel, J. VanderPlas, A. Granfort, F. Pedregosa, A. Mueller, and G. Louppe (2016), scikitlearn, http://mloss.org/software/view/240/.

Tibshirani, R. (1996), Regression shrinkage and selection via the lasso, *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288.

Ting, M. H., C. M. Chu, G. Zeng, D. Li, and G. Chng (2017), Predicting recidivism among youth offenders: Augmenting professional judgement with machine learning algorithms, *Journal of Social Work*, pp. 631–649.

Tollenaar, N., and P. Van der Heijden (2013), Which method predicts recidivism best?: a comparison of statistical, machine learning and data mining predictive models, *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, *176*(2), 565–584.

Townsend, J. T. (1971), Theoretical analysis of an alphabetic confusion matrix, *Perception & Psychophysics*, *9*(1), 40–50.

Tribby, C. P., H. J. Miller, B. B. Brown, C. M. Werner, and K. R. Smith (2017), Analyzing walking route choice through built environments using random forests and discrete choice techniques, *Environment and Planning B: Urban Analytics and City Science*, *44*(6), 1145–1167.

Ursenbach, J., J. Neiser, M. E. O'Connell, and R. J. Spiteri (2018), Alternative scoring algorithm for a computer-based cognitive screening tool, unpublished.

Williams, N., S. Zander, and G. Armitage (2006), A preliminary performance comparison of five machine learning algorithms for practical ip traffic flow classification, *ACM SIGCOMM Computer Communication Review*, *36*(5), 5–16.

Wold, H. (2004), Partial least squares, *Encyclopedia of statistical sciences*, *9*, 581–591.

Wormith, J. S. (2011), The legacy of D. A. Andrews in the field of criminal justice: How theory and research can change policy and practice, *International Journal of Forensic Mental Health*, *10*(2), 78–82.

Xu, J., B. Xu, W. Zhang, Z. Cui, and W. Zhang (2007), A new feature selection method for text clustering, *Wuhan University Journal of Natural Sciences*, *12*(5), 912–916.

Yang, Q., T. Li, and K. Wang (2003), Web-log cleaning for constructing sequential classifiers, *Applied Artificial Intelligence*, *17*(5-6), 431–441.

Zamir, O., and O. Etzioni (1998), Web document clustering: A feasibility demonstration, in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 46–54, ACM.

Zhang, S., C. Zhang, and Q. Yang (2003), Data preparation for data mining, *Applied artificial intelligence*, *17*(5-6), 375–381.

Zhu, W., N. Zeng, N. Wang, et al. (2010), Sensitivity, specificity, accuracy, associated confidence interval and ROC analysis with practical SAS implementations, *NESUG proceedings: health care and life sciences, Baltimore, Maryland*, *19*, 67.

Appendix A

Histograms of all Features

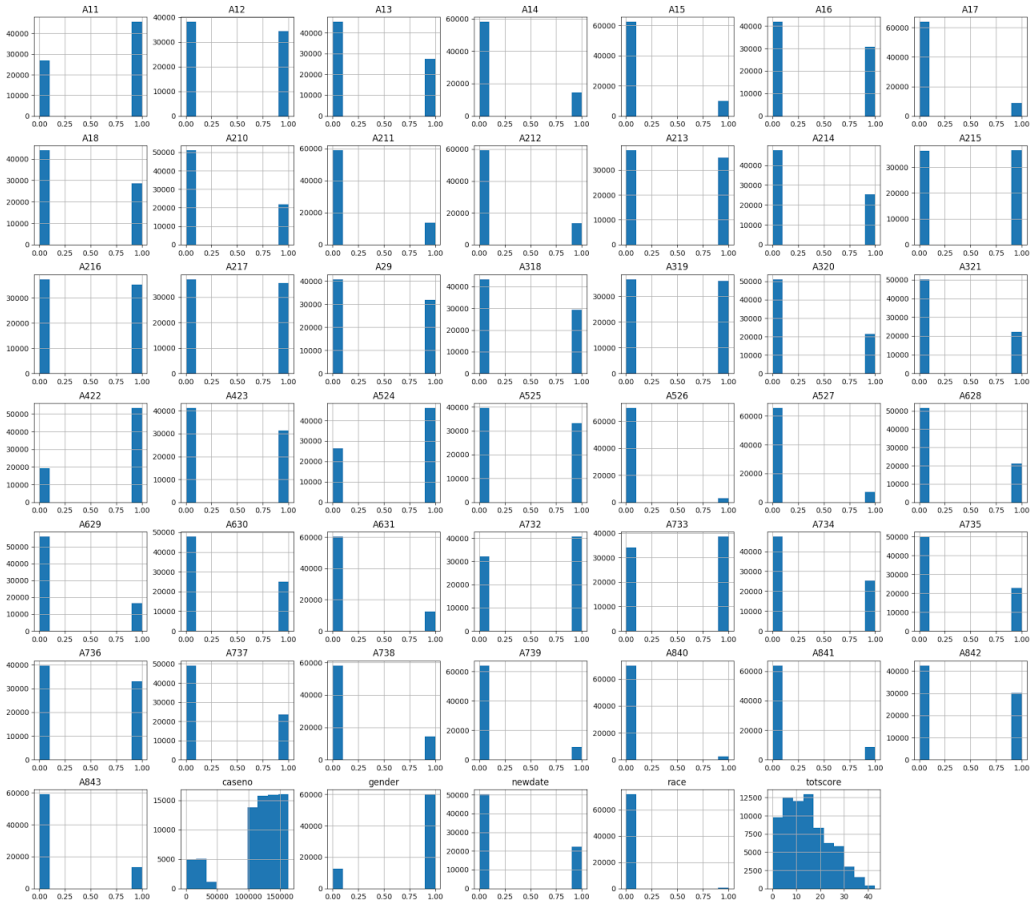

Figure A.1: Histograms of all Features

Appendix B

CAMCI Variables (*Saxton et al.*, 2009)

The following test descriptions are from *Saxton et al.* (2009) and are what make up the CAMCI test. The variables in the data set correspond to one of the below tests. The names of the variables are:

- BaselineStatClas
- Age
- YrsEdu
- SRAllRspDRT
- SRCorRspDRT
- SRNmCorTGAC
- SRNmIncRspAC
- RP1CorRspART
- RPCorRspDRT
- RPDPrimeAC
- RPIncRspART
- RPNmCorRspAC
- RPNmIncRspAC
- WRNmCorRspAC
- WRNmOmAC
- GN1CorRspART
- GN1NmCorAC
- GN2CorRspART
- GN2NmCorAC
- DF1stTrlRT
- DFMaxSpanAC
- DR1stTrlRT
- DRMaxSpanAC
- VRAllRspART
- VRCorRspDRT
- VRNmCorRspAC
- VEATMAC
- VEINNmCorAC
- VEINTAC
- VEITNmCorAC
- RP3CorRspART
- RP4CorRspART

Modified Pencil-and-Paper Tests for Computer Presentation

Star Task (Attention): The participants were shown a star, circle, square, or triangle and told to tap the screen as quickly as possible only when the star appeared.

Forward Digit Span (Attention): A series of 3 to 6 numbers was presented, 1 per second. After presentation, participants were told to recall the numbers in the correct order using a display at the bottom of the screen.

Word Recognition (Verbal Memory): A list of 6 words was shown on the screen, 1 at a time. The participant was instructed to remember each word and warned that they would be asked to recall it later. After a delay, 6 sets of 4 words (3 distracters and the target word) were displayed and the task was to tap the target word.

Word Recall (Verbal Memory): Participants were presented with a series of five 3-letter words one at a time and told to remember them. The words were presented 3 times. Approximately 10 minutes later participants were asked to recall the 3-letter words by typing them on a keyboard that appeared at the bottom of the screen.

Picture Recognition (Visual Memory): The participant was shown a fixed series of pictures. Some had been shown previously and some were new. The participant was told to tap "yes" if the picture had been shown previously and "no" if it had not.

Go/No-Go Test (Executive Function): In Part 1 of this task, participants were asked to tap the screen twice when they heard 1 beep and once when they heard 2 beeps. In Part 2 the rules were changed and participants were told to tap twice when they heard 1 beep and do nothing when they heard 2 beeps.

Digit Reverse Span (Working Memory) : This task is identical to Forward Digit Span except that participants were told to recall the numbers in reverse order.

CAMCI Shopping Trip/Virtual Reality

Prospective Memory: At the beginning of the shopping trip the participant was told that on the way to the market they must stop at the bank to transfer money and at the post office to mail a letter. The participant was told to tap on the image of the bank and the post office when they appeared on the screen to indicate that they had remembered to perform these tasks.

Shopping Trip Choice Points: At the beginning of the shopping trip the participant is provided with directions to Sullivan's Market. The directions include statements such as "Make a left on Fir Street, then make a right on Ash Street." The directions remained on the screen throughout the test so the participant could refer to them at any time. At each intersection the participant had to decide which direction to go.

Bank Machine: the participant arrived at the bank he/she saw a standard bank automatic teller machine (ATM). The participant was told to transfer $250.00 from the savings account to the checking account. Each step of the transaction was scored for correctness and time taken to complete the task. If the participant did not remember to stop at the bank, the CAMCI automatically "drove" to the bank so that this portion of the test was completed by all participants.